National Library
of Canada

Acquisitions and
Bibliographic Services Branch

395 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale
du Canada

Direction des acquisitions et
des services bibliographiques

395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Your file Votre référence

Our file Notre référence

NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30, and subsequent amendments.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30, et ses amendements subséquents.

Canada

UNIVERSITY OF ALBERTA

UNIFIED MODEL PREDICTIVE CONTROL

BY

MUNAWAR SAUDAGAR ©

A THESIS
SUBMITTED TO THE FACULTY OF GRADUATE STUDIES AND RESEARCH IN
PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY
IN
PROCESS CONTROL

DEPARTMENT OF CHEMICAL ENGINEERING

EDMONTON, ALBERTA
FALL, 1995

National Library
of Canada

Acquisitions and
Bibliographic Services Branch

395 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale
du Canada

Direction des acquisitions et
des services bibliographiques

395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Your file Votre référence

Our file Notre référence

THE AUTHOR HAS GRANTED AN IRREVOCABLE NON-EXCLUSIVE LICENCE ALLOWING THE NATIONAL LIBRARY OF CANADA TO REPRODUCE, LOAN, DISTRIBUTE OR SELL COPIES OF HIS/HER THESIS BY ANY MEANS AND IN ANY FORM OR FORMAT, MAKING THIS THESIS AVAILABLE TO INTERESTED PERSONS.

L'AUTEUR A ACCORDE UNE LICENCE IRREVOCABLE ET NON EXCLUSIVE PERMETTANT A LA BIBLIOTHEQUE NATIONALE DU CANADA DE REPRODUIRE, PRETER, DISTRIBUER OU VENDRE DES COPIES DE SA THESE DE QUELQUE MANIERE ET SOUS QUELQUE FORME QUE CE SOIT POUR METTRE DES EXEMPLAIRES DE CETTE THESE A LA DISPOSITION DES PERSONNE INTERESSEES.

THE AUTHOR RETAINS OWNERSHIP OF THE COPYRIGHT IN HIS/HER THESIS. NEITHER THE THESIS NOR SUBSTANTIAL EXTRACTS FROM IT MAY BE PRINTED OR OTHERWISE REPRODUCED WITHOUT HIS/HER PERMISSION.

L'AUTEUR CONSERVE LA PROPRIETE DU DROIT D'AUTEUR QUI PROTEGE SA THESE. NI LA THESE NI DES EXTRAITS SUBSTANTIELS DE CELLE-CI NE DOIVENT ETRE IMPRIMES OU AUTREMENT REPRODUITS SANS SON AUTORISATION.

ISBN 0-612-06280-5

Canada

UNIVERSITY OF ALBERTA
RELEASE FORM

Name of Author: Munawar Saudagar

Title of Thesis: Unified Model Predictive Control

Degree: Doctor of Philosophy

Year this Degree Granted: 1995

Permission is hereby granted to the University of Alberta Library to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly, or scientific research purposes only.

The author reserves all other publication and other rights in association with the copyright in the thesis, and except as hereinbefore provided, neither the thesis nor any substantial portion thereof may be printed or otherwise reproduced in any material form without the author's prior written permission.

508 RH, Michener Park
Edmonton, Alberta T6H 4M5

Date: October 3, 1995

UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled **Unified Model Predictive Control** submitted by Munawar Saudagar in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Process Control.

Dr. D.G. Fisher
(Supervisor)

Dr. Ming Rao
(Co-supervisor)

Dr. S.L. Shah
(Chemical Engineering)

Dr. R.K. Wood
(Chemical Engineering)

Dr. M. Meng
(Electrical Engineering)

Dr. D.W. Clarke
(Oxford University)

Date: September 22, 1995

Say: "Truly, my prayer and my sacrifice,
my life and my death are all
Dedicated to God,
The Cherisher of the Worlds"

(*Quran 6-162*)

Abstract

The distinguishing features of Model Predictive Controllers (MPC's) include a process plus noise *model*, a *predictor* to produce estimates of the future output trajectory, a *control algorithm* to calculate the future control trajectory that minimizes a user specified performance index, and a large number of tuning parameters which provide the flexibility required to handle a wide range of applications. The main contributions of this thesis in these areas are:

- A generalized Box-Jenkins type *model* that unifies the development of most discrete equation error and output error formulations including GPC and DMC. The classical step response form of DMC is derived from the same generalized transfer function model as the GPC type controllers and extended to include a lead/lag noise model that can be used to speed up or filter the disturbance rejection independent of the servo response.

- A Separated Diophantine Overparameterized *Predictor* (SDOP) for stable processes that separates the process and noise model contributions to the process output plus a Reduced Diophantine Overparameterized Predictor (RDOP) that reduces adaptive control calculations by up to 65%. Both are proven to be equivalent to the optimal predictors used in GPC.

- A *control algorithm* to calculate the vector of control moves that minimizes a quadratic performance index containing the squares of the control errors and control increments over a "sparse" set of *selected* future time instants. A normalized μ-weighting parameter is incorporated so that varying μ from zero to

unity produces the effect of *continuously* varying the control horizon from N_u to N_u+m ($m \geq 1$).

- To facilitate *tuning* of the overall (servo) response the predicted steady state output is added as the last element in the predicted output vector and it is shown that varying the normalized steady state weighting parameter, $0 \leq \gamma_s \leq 1$, varies the closed-loop response from very aggressive (in the extreme *deadbeat*) to a very conservative *mean level* response.

- Several alternatives for *tuning* the regulatory calculation (disturbance rejection) properties of MPC's are derived based on the generalized noise model $C / D\breve{A}\Delta^n$. The recommended approach is to use a "disturbance horizon " or a normalized, noise-model weighting factor, $0 \leq \sigma \leq 1$, which varies the contribution of the noise model to the output response (independent of the servo response) from conservative (e.g. DMC) to aggressive (e.g. GPC). Adjustment of the normalized σ-weighting (for uncertainty prediction) and γ_s-weighting (for overall performance) is sufficient to tune most practical applications.

These new developments are combined in a Unified Model Predictive Controller (UMPC) that is attractive for theoretical work because of its generality and for practical applications because of its flexibility and ease of tuning.

Acknowledgment

All praise and gratitude are due to God, my creator and sustainer who enabled me to accomplish this research work.

I am grateful to my supervisor, Dr. Grant Fisher, for his technical advice, personal guidance and constant encouragement during the course of this research. I am also thankful to Dr. Sirish Shah, Dr. Ming Rao and Dr. R.K. Wood for their personal assistance.

I would like to extend my thanks to the graduate students in Process Control for many hours of technical discussion and useful comments regarding my research work. I am also grateful to other graduate students and freinds for their moral support, friendship and cooperation.

Financial support from the Department of Chemical Engineering and the Natural Sciences and Engineering Research Council is gratefully acknowledged.

My special thanks and gratitude are due to my late parents for their unlimited love, care and support. May God bless their souls and have mercy on them. I am also grateful to my brother and sisters who encouraged me in my pursuit of higher education.

Finally I would like to thank my wife Samira and children Mariam, Mustafa and Ibrahim for their constant support, sacrifice and patience throughout my Ph.D. program.

Contents

1 Introduction ---------- 1

1.1 Model Predictive Control (MPC) ---------- 1

1.1.1 Dynamic Matrix Control (DMC) ---------- 4

1.1.2 Model Algorithmic Control / Identification and Command (MAC / IDCOM) ---------- 4

1.1.3 Generalized Predictive Control (GPC) ---------- 5

1.1.4 Commercial MPC Packages ---------- 6

1.2 Advantages of MPC ---------- 8

1.3 Drawbacks of MPC ---------- 8

1.4 Structure of the Thesis ---------- 9

1.4.1 Motivation and Objectives ---------- 9

1.4.2 Thesis Organization ---------- 11

References ---------- 13

2 Unified Model Predictive Control ---------- 15

2.1 Introduction ---------- 15

2.2 The Generalized Model Structure ---------- 16

2.3 The Lumped Diophantine Predictor (LDP) ---------- 22

2.4 The Lumped Diophantine Overparameterized Predictor (LDOP) ---------- 26

2.5 The Control Law ---------- 30

2.6 Improved Dynamic Matrix Control (IDMC) ---------- 33

2.6.1 Classical DMC Predictor using LDP ---------- 34

2.6.2 Improved DMC (IDMC) Predictor using LDP ---------- 36

2.7 Simulation Examples ---------- 40

2.7.1 Example-1: Effect of Model Structure ---------- 41

2.7.2 Example-2: Advantages of using an additional factor D in the Denominator of the Noise Model ---------------------------------42

2.7.3 Example-3: Double Integrator for Ramp disturbances-------------------42

2.7.4 Example-4: Double Integrator for Ramp Set point Tracking------------43

2.7.5 Example-5: Disturbance Rejection Properties of IDMC -----------------43

2.8 Conclusions---44

References --45

3 Disturbance Rejection Techniques for Long Range Predictive Control----------57

3.1 Introduction---57

3.2 Model and Predictor Preliminaries ---58

3.3 Separated Diophantine Predictor (SDP) --60

3.4 The Separated Diophantine Overparameterized Predictor (SDOP)------------68

3.5 The Control Law---72

3.6 Enhanced Dynamic Matrix Control (EDMC) -------------------------------------74

3.6.1 Classical DMC Predictor using SDP ---------------------------------------74

3.6.2 Enhanced DMC (EDMC) Predictor using SDP ------------------------76

3.7 Disturbance Horizon---78

3.8 σ-Weighting ---81

3.9 Simulation Examples---82

3.9.1 Example-1: DMC and EDMC Performance Comparison----------------83

3.9.2 Example-2: Demonstration of Disturbance Horizon as a Tuning Parameter ---83

3.9.3 Example-3: Multiple Noise Models using σ-Weighting------------------84

3.10 Conclusions---85

References --86

Appendix 3-A Comparison of UMPC with the Unified Predictive Control (UPC) of Soeterboek (1990)--88

4 Reduced Diophantine Predictors for Long Range Predictive Control---------- 107

4.1 Introduction-- 107

4.2 The Reduced Diophantine Predictor (RDP) ---------- 110
4.3 The Reduced Diophantine Overparameterized Predictor (RDOP) ---------- 116
4.4 The Control Law ---------- 120
4.5 Conclusions ---------- 123
References ---------- 123
Appendix 4-A Polynomial Shift Algebra ---------- 124

5 Generalized Cost Function for Long Range Predictive Control ---------- 128
5.1 Introduction ---------- 129
5.2 Output Horizon Restructuring ---------- 131
5.2.1 Sparse Output Horizon ---------- 131
5.2.2 Integral Error Cost ---------- 134
5.3 Generalized Control Horizon ---------- 135
5.3.1 Planned control Policy (PCP) ---------- 135
5.3.2 The d_2 Spacing ---------- 136
5.3.3 The d_3 Spacing ---------- 137
5.4 Multiple Control Horizons ---------- 138
5.4.1 Control Horizons 1 and N_u ---------- 139
5.4.2 Control Horizons N_u-1 and N_u ---------- 139
5.5 Simulation Examples ---------- 140
5.5.1 Example-1: Sparse Output Horizon ---------- 141
5.5.2 Example-2: Sparse Output Horizon with Integral Cost ---------- 141
5.5.3 Example-3: Interpolating Effect of d_2 ---------- 142
5.5.4 Example-4: Interpolating Effect of d_3 ---------- 142
5.5.5 Example-5: Effect of Multiple Control Horizons ---------- 142
5.6 Conclusions ---------- 143
References ---------- 144

6 Unified Predictive Control with Steady State Error Weighting ---------- 157
6.1 Introduction ---------- 157

6.2 The Generalized Measured Output Model and the Long Range Predictor 160
6.3 Optimal Steady State Prediction 163
  6.3.1 The Steady State Prediction Method-I 164
  6.3.2 The Steady State Prediction Method-II 166
6.4 Pure Steady State Control for Finite Control Horizon 167
6.5 Combination of Steady State Control with LRPC 169
6.6 Disturbance Modeling for Steady State Prediction 170
6.7 Simulation Examples 173
  6.7.1 Example-1: Detuning Effect of γ_s-Weighting 174
  6.7.2 Example-2: γ_s-Weighting to mimic a larger N_2 174
  6.7.3 Example-3: Use of DMC noise model for steady state prediction 175
  6.7.4 Example-4: Comparison between λ and γ_s-Weightings 176
6.8 Conclusions 176
References 177
Appendix 6-A Review of the Existing Formulation for GPC with Steady State Error Weighting 178

7 Interpolation and Recursion of Control Horizon in Long Range Predictive Control 192
7.1 Introduction 193
7.2 Review of Conventional Long Range Predictive Control Algorithm 193
7.3 Control Horizon Interpolation Method-I 196
7.4 Control Horizon Interpolation Method-II 197
7.5 Control Horizon Recursion 198
  7.5.1 Recursive Formulation 198
  7.5.2 Recursive Implementation 200
7.6 Control Horizon Interpolation with Recursion 201
7.7 Simulation Examples 204
  7.7.1 Example-1: μ-interpolation of control horizon, Method-I 204
  7.7.2 Example-2: μ-interpolation of control horizon, Method-2 205

7.7.3 Example-3: Recursion of control horizon ------------------------------ 205
7.7.4 Example-4: Recursion with interpolation of control horizon ---------- 205
7.8 Conclusions -- 206
References --- 206

8 Conclusions -- 215
8.1 Contributions -- 215
8.1.1 Generalized Noise/Disturbance Model Structure ------------------------ 215
8.1.2 Optimal Long-Range Predictors -- 216
8.1.3 Use of Multiple Noise/Disturbance Models ----------------------------- 217
8.1.4 Steady State Error Weighting --- 219
8.1.5 Horizon Restructuring and Interpolation ------------------------------ 220
8.1.5.1 Structuring the Output Horizon ------------------------------------ 220
8.1.5.2 Structuring the Control Horizon ----------------------------------- 220
8.1.6 Interpolation and Recursion of the Control Horizon ------------------ 221
8.2 Tuning Guidelines for UMPC -- 222
8.2.1 UMPC for Open-loop Stable Processes --------------------------------- 222
8.2.2 UMPC for Open-loop Unstable Processes ------------------------------- 223
8.3 Future Work -- 223
References --- 225

List of Figures

2.1 Unified model predictive control (UMPC) with lumped (GPC type) Diophantine predictor (LDP) ----------47

2.2 Analysis of LDP to separate free responses corresponding to the manipulated variable and the uncertainty ----------48

2.3 Effect of noise model in the presence of model plant mismatch (MPM) -----------49

2.4 Effect of noise model in the presence of MPM for Rohr' process -----------------50

2.5 Role of D polynomial in disturbance rejection properties of UMPC --------------51

2.6 Effect of D polynomial in the presence of MPM -----------------------------------52

2.7 Offset removal using double integrator for ramp disturbances in DMC -----------53

2.8 Offset removal using double integrator for ramp disturbances in GPC ------------54

2.9 Offset-free ramp tracking using a double integrator --------------------------------55

2.10 Disturbance rejection properties of improved DMC (IDMC) ----------------------56

3.1 Unified model predictive control (UMPC) with separated Diophantine predictor (SDP) ----------91

3.2 Uncertainty projection at various frequencies for GPC noise model --------------92

3.3 Uncertainty projection at various frequencies for GPC with T-filter --------------93

3.4 Uncertainty projection for DMC with a first order lag filter -----------------------94

3.5 Uncertainty projection for various noise models at a fixed frequency -------------95

3.6 Uncertainty projection using disturbance horizon for GPC followed by DMC ---96

3.7 Uncertainty projection using disturbance horizon for DMC followed by GPC ---97

3.8 Uncertainty projection using the limited expansion of GPC noise model ---------98

3.9 σ-Weighting between DMC and GPC noise models --------------------------------99

3.10 Disturbance rejection using enhanced DMC (EDMC) ---------------------------- 100

3.11 Effect of C polynomial on noise filtering in enhanced DMC (EDMC) ---------- 101

3.12 Incorporating two noise models (GPC & DMC) using disturbance horizon ---- 102

3.13 Incorporating two noise models (DMC & GPC) using disturbance horizon ---- 103

3.14 Limiting the GPC noise model expansion up to a disturbance horizon ---------- 104

3.15 σ-Weighting to incorporating two noise models (GPC & DMC) 105

3.16 σ-Weighting to incorporating two noise models in the presence of MPM 106

4.1 Unified model predictive control (UMPC) with reduced Diophantine predictor (RDP) 126

4.2 Analysis of RDP to separate free responses corresponding to the manipulated variable and the uncertainty 127

5.1 Concept of a conventional long-range predictive controller 145

5.2 Long-range predictive controller with sparse output horizon ($d_1 = 3$) 146

5.3 Long-range predictive controller with generalized control horizon, ($d_2 = 2$) 147

5.4 Long-range predictive controller with generalized control horizon, ($d_3 = 1$) 148

5.5 LRPC with regular and sparse output horizons 149

5.6 Integral and summation costs for LRPC with sparse output horizon 150

5.7 LRPC with control horizons 1, 2, and 3 151

5.8 Interpolating effect of d_2 152

5.9 Interpolating effect of d_3 for $N_u = 3$ 153

5.10 Interpolating effect of d_3 for $N_u = 2$ 154

5.11 Multiple control horizons, $N_u = 1$ and $N_u = 3$ 155

5.12 Multiple control horizons, $N_u = 2$ and $N_u = 3$ 156

6.1 Incorporation of steady state error in long-range predictive control 181

6.2 Detuning with steady state error weighting for process A, $\gamma_s = 0$ to 0.2 182

6.3 Detuning with steady state error weighting for process A, $\gamma_s = 0.3$ to 0.9 183

6.4 Detuning with steady state error weighting for process B, $\gamma_s = 0$ to 0.2 184

6.5 Detuning with steady state error weighting for process B, $\gamma_s = 0.3$ to 0.9 185

6.6 A small N_2 plus γ_s gives the effect of larger N_2, DMC for process A 186

6.7 A small N_2 plus γ_s gives the effect of larger N_2, GPC for process A 187

6.8 A small N_2 plus γ_s gives the effect of larger N_2, DMC for process A with sever model plant mismatch (MPM) 188

6.9 Effect of noise model structure on steady state prediction calculation 189

6.10 γ_s-weighting vs. λ-weighting, GPC for process B with MPM 190

6.11 γ_s-weighting vs. λ-weighting, GPC for process A with MPM 191

7.1 μ-weighting interpolation between $N_u = 1$ and $N_u = 2$, Method-I ---------------- 208

7.2 μ-weighting interpolation between $N_u = 1$ and $N_u = 3$, Method-I ---------------- 209

7.3 μ-weighting Methods I and II, $N_u = 1$, $m = 1$, and $\mu = 0.8$ ------------------------ 210

7.4 Non recursive $N_u = 2$ vs. recursive $N_u = 1$ with $m = 1$ ------------------------------ 211

7.5 Non recursive $N_u = 3$ vs. recursive $N_u = 1$ with $m = 2$ ------------------------------ 212

7.6 Non recursive $N_u = 3$ vs. recursive $N_u = 2$ with $m = 1$ ------------------------------ 213

7.7 m-interpolation without and with recursion, $N_u = 1$, $m = 1$ and $\mu = 0.8$ --------- 214

List of Symbols

$A(q^{-1})$ or A	Process model denominator polynomial
$\breve{A}$	A factor that becomes A for equation error models and 1 for output error models
$\overset{\frown}{A}$	A factor that becomes A for output error models and 1 for equation error models
A_s	Stable part of polynomial A
A_u	Unstable part of polynomial A
$B(q^{-1})$ or B	Process model numerator polynomial
$C(q^{-1})$ or C	Noise model numerator polynomial
$D(q^{-1})$ or D	A factor (polynomial) in the noise model denominator
d	Time delay in sample intervals
d_1	Output horizon spacing
d_2	Control horizon spacing, between first two control moves
d_3	Control horizon spacing, between last two control moves
d_4	Separating point, in output horizon, for two control horizons
E_j	Future polynomial in j-step ahead Diophantine of noise model, $\dfrac{C}{D\breve{A}\Delta^n}$
E_N	Future polynomial in N-step ahead Diophantine of noise model, $\dfrac{C}{D\breve{A}\Delta^n}$
$\widetilde{E}_{Nj}$	First j terms of E_N
$\overline{E}_{Nj}$	Last N-j terms of E_N
$e(t)$	Fundamental uncertainty generating signal (e.g. white noise)
$e^f(t)$	$e(t)$ filtered by $\dfrac{1}{D\breve{A}\Delta^n}$
e_s	Steady state value of e_j

F_j	Past polynomial in *j*-step ahead Diophantine of the noise model, $\frac{C}{D\breve{A}\Delta^n}$
F_N	Past polynomial in *N*-step ahead Diophantine of the noise model, $\frac{C}{D\breve{A}\Delta^n}$
F_s	Steady state value of F_j
$\mathbf{f}$	Free response vector
$\mathbf{f}^{\#}$	Vector formed by stacking **f** on itself
$f(t+j)$	Free response at $t+j$
$\mathbf{G}$	Dynamic matrix , the matrix of $\tilde{G}_j$ elements
$\mathbf{G_m}$	Matrix containing N_u+1 to N_u+m columns of **G**
$\mathbf{G_{+m}}$	Matrix formed by adjoining $\mathbf{G_m}$ with **G**, $[\mathbf{G}\ \ \mathbf{G_m}]$
$\mathbf{G_{+0}}$	Matrix formed by adjoining a zero matrix of $\mathbf{G_m}$ dimension with **G**, $[\mathbf{G}\ \ \mathbf{0}]$
$\mathbf{G}^{\#}$	Matrix formed by stacking $\mathbf{G_{+m}}$ and $\mathbf{G_{+0}}$
$G(q^{-1})$	Process model
$\tilde{G}_j$	Future polynomial in *N*-step ahead Diophantine expansion of $\frac{E_N DB}{C\hat{A}}$
$\overline{G}_j$	Past polynomial in *j*-step ahead Diophantine expansion of $\frac{E_j DB}{C\hat{A}}$
$\tilde{G}_N$	Future polynomial in *j*-step ahead Diophantine expansion of $\frac{E_j DB}{C\hat{A}}$
$\overline{G}_N$	Past polynomial in *N*-step ahead Diophantine expansion of $\frac{E_N DB}{C\hat{A}}$
$\tilde{\tilde{G}}_{Nj}$	First *j* terms of $\tilde{G}_N$
$\overline{\tilde{G}}_{Nj}$	Last *N-j* terms of $\tilde{G}_N$
g_i	Coefficients of $\tilde{G}_j$
g_s	Steady state value of g_i
H	Polynomial of impulse response coefficients
$\tilde{H}_j$	Future polynomial in *j*-step ahead Diophantine of $\frac{B}{A}$, first *j* terms of H

$\overline{H}_j$	Past polynomial in j-step ahead Diophantine of $\frac{B}{A}$
h_i	Impulse response coefficients (coefficients of H)
J	Quadratic cost function (performance index)
J_e	Error cost, the sum of square of weighted tracking (control) errors
J_c	Control cost, the sum of square of weighted control moves
	cost function J
N	Number of tracking (control) errors in J_e
N_d	Disturbance horizon
N_u	Control horizon, the number of future non-zero control increments
N_1	Initial output horizon ≥ the earliest future output that is affected by the control move at time t
N_2	Final output horizon, the farthest future output that is included in the n
	Number of integrators in the noise model
$\tilde{P}_j$	Future polynomial in j-step ahead Diophantine of $\frac{B}{\Delta^n A}$
$\tilde{P}_N$	Future polynomial in N-step ahead Diophantine of $\frac{B}{\Delta^n A}$
$\tilde{\tilde{P}}_{Nj}$	First j terms of $\tilde{P}_N$
$\overline{\tilde{P}}_{Nj}$	First N-j terms of $\tilde{P}_N$
$\overline{P}_j$	Past polynomial in j-step ahead Diophantine of $\frac{B}{\Delta^n A}$
$\overline{P}_N$	Past polynomial in N-step ahead Diophantine of $\frac{B}{\Delta^n A}$
$\overline{P}_s$	Steady state value of $\overline{P}_j$
q^{-1}	Backward shift operator
S	Polynomial of step response coefficients
$\tilde{S}_j$	First j terms of S
$\overline{S}_j$	S with first j terms removed and shifted by q^j

s_i	Step response coefficients (coefficients of S)
T	Process settling time
$\mathbf{u}$	Vector of current and future control moves
$u(t)$	Process input or controller output
$u^f(t)$	Process input or controller output filtered by $\frac{1}{C\hat{A}}$
$u^F(t)$	Process input or controller output filtered by $\frac{1}{A}$
$\tilde{V}_j$	Future polynomial in *j*-step ahead Diophantine of $\frac{1}{DA\Delta^n}$
$\tilde{V}_N$	Future polynomial in *N*-step ahead Diophantine of $\frac{1}{DA\Delta^n}$
$\bar{V}_j$	Past polynomial in *j*-step ahead Diophantine of $\frac{1}{DA\Delta^n}$
$\bar{V}_N$	Past polynomial in *N*-step ahead Diophantine of $\frac{1}{DA\Delta^n}$
$\left[\tilde{V}_j C\hat{A}\right]_j$	Future polynomial in *j*-step ahead Diophantine of $\tilde{V}_j C\hat{A}$
$\left[\tilde{V}_N C\hat{A}\right]_j$	Future polynomial in *j*-step ahead Diophantine of $\tilde{V}_N C\hat{A}$
$\overline{\left[\tilde{V}_j C\hat{A}\right]}_j$	Past polynomial in *j*-step ahead Diophantine of $\tilde{V}_j C\hat{A}$
$\overline{\left[\tilde{V}_N C\hat{A}\right]}_j$	Past polynomial in *j*-step ahead Diophantine of $\tilde{V}_N C\hat{A}$
$\left[\tilde{V}_j DB\right]_j$	Future polynomial in *j*-step ahead Diophantine of $\tilde{V}_j DB$
$\left[\tilde{V}_N DB\right]_j$	Future polynomial in *j*-step ahead Diophantine of $\tilde{V}_N DB$
$\overline{\left[\tilde{V}_j DB\right]}_j$	Past polynomial in *j*-step ahead Diophantine of $\tilde{V}_j DB$
$\overline{\left[\tilde{V}_N DB\right]}_j$	Past polynomial in *j*-step ahead Diophantine of $\tilde{V}_N DB$
$W(q^{-1})$ or W	Noise model
$\mathbf{w}$	Vector of future setpoints
$\mathbf{w}^{\#}$	Vector formed by stacking **w** on itself

$w(t)$	the desired output value or the setpoint
$x(t)$	Residual, the difference in process measured and model outputs
$x^f(t)$	Filtered residual, $x(t)$ filtered by $\frac{1}{C}$
$y(t)$	Measured output
$y^f(t)$	Measured output filtered by $\frac{1}{C}$
$y_m(t)$	Model output
$\mathbf{y_p}$	Vector of predicted process output values
$y_p(t+j\|t)$	The j-step ahead optimal output prediction conditioned on the input/output data up to time t
$y_p(t+j\|t,N_u)$	The j-step ahead optimal output prediction conditioned on the input/output data up to time t and based on a control horizon of N_u .

Greek Symbols

Δ	Differencing operator, $1-q^{-1}$
Γ	Matrix of tracking error weightings
$\Gamma^{\#}$	Matrix of tracking error weightings corresponding to $\mathbf{G}^{\#}$
Λ	Matrix of control weightings
$\Lambda^{\#}$	Matrix of control weightings corresponding to $\mathbf{G}^{\#}$
Λ_m	Matrix of control weightings corresponding to $\mathbf{G_m}$
Θ	Numerator polynomial for model following
Ω	Denominator polynomial for model following
Ψ_j	A factor, $\frac{1-\delta^{j+1}}{1-\delta}$, used in IDMC
χ	Coefficient of the first order C polynomial used in EDMC
δ	Coefficient of the first order D polynomial used in IDMC and EDMC
$\varepsilon_p(t+j\|t)$	Prediction error

γ	Weighting on tracking error.
γ_s	Steady state control error weighting, normalized
λ	Control weighting
μ	A normalized parameter that gives the relative contribution of a control horizon to the output predictions
σ	A normalized parameter that gives the relative contribution of a noise model to the output predictions

Abbreviations

ARIMAX	Auto Regressive Integrated Moving Average with eXogenous input
ARIMA	Auto Regressive Integrated Moving Average
ARMA	Auto Regressive Moving Average
ARMAX	Auto Regressive Moving Average with eXogenous input
ARIX	Auto Regressive Integrated with eXogenous input
ARX	Auto Regressive with eXogenous input
BJ	Box-Jenkins
CARIMA	Controlled Auto regressive integrated moving average
DMC	Dynamic Matrix Control
DMI	Dynamic Matrix Identification
EDMC	Enhanced Dynamic matrix control
EE	Equation error (a class of model structure)
EHAC	Extended Horizon Adaptive Control
ESPAC	Extended Predictive Self Adaptive Control
GPC	Generalized Predictive Control
HMPC	Honeywell Multivariable Predictive Control
HPC	Horizon Predictive Control
IDCOM	IDentification and COMmand
IDMC	Improved Dynamic Matrix Control
KF	Kalman Filter

LDP	Lumped Diophantine Predictor
LDOP	Lumped Diophantine Overparameterized Predictor
LRPC	Long-Range Predictive Control(ler)
MAC	Model Algorithmic Control
MBPC	Model Based Predictive Control
MIMO	Multi-Input Multi-Output
MPC	Model Predictive Control
MPM	Model Plant Mismatch
MURHAC	Multipredictor Receding Horizon Adaptive Control
MUSMAR	MUltiStep Multivariable Adaptive Control
OCS	Optimum Control Synthesis
OPC	Optimum Predictive Control
PC	Predictive Control
PCP	Planned Control Policy
PID	Proportional Integral Derivative
QDMC	Quadratic Dynamic Matrix Control
RDP	Reduced Diophantine Predictor
RDOP	Reduced Diophantine Overparameterized Predictor
RTO	Real-Time Optimization
SDP	Separated Diophantine Predictor
SDOP	Separated Diophantine Overparameterized Predictor
SISO	Single-Input Single-Output
UPC	Unified Predictive Control
UMPC	Unified Model Predictive Control

Chapter 1

Introduction

The results presented in this thesis are related to the increasingly popular process control strategy known as Model Predictive Control or MPC. MPC is receiving widespread acceptance both in industry and in academia. In order to establish a proper perspective for the research work reported in this thesis various aspects of MPC are outlined in the following sections.

1.1 Model Predictive Control (MPC)

Model Predictive Control (MPC), which is sometimes also referred to as Model Based Predictive Control (MBPC) or just Predictive Control (PC), is a subset of Model-Based Control (MBC) that has been defined differently by various researchers. Froisy (1994) defines MPC to be the control approaches that use embedded process models. Muske and Rawlings (1993) view linear MPC as a class of control algorithms that compute a manipulated variable profile by utilizing a linear process model to optimize a linear or quadratic open-loop performance function subject to linear constraints over a future time horizon. The first move of this open-loop optimal manipulated variable profile is then implemented. This procedure is repeated at each control interval with the process measurements used to update the optimization problem. According to Ogunnaike and Ray (1994) MPC is an appropriately descriptive name for a class of computer control schemes that utilize a process model for two central tasks:

- Explicit prediction of future plant behavior.
- Computation of the corrective control action required to derive the predicted output as close as possible to the desired target values.

According to Richalet (1992), a typical MPC involves the following four activities:

- *Training* or operating image — Process model
- *Target* — Reference trajectory
- *Action* — Computation of a structured manipulated variable
- *Comparison* actual vs. expected — Modeling error compensator

De Keyser (1991) summarizes some of the important elements characterizing MPC as follows:

- Prediction by means of a process model.
- Generation of a reference trajectory.
- Structuring of the (postulated) control law.
- Algorithmic calculation of the best control scenario.

An important MPC characteristic, common to most of the existing algorithms, is the use of future trajectories of open-loop plant output as a basis for calculating the future control horizon. Based on this concept MPC is also called Long Range Predictive Control (LRPC).

Dynamic Matrix Control (DMC) and Model Algorithmic Control (MAC), which is also known as IDCOM, were the pioneering industrial applications which in fact established the area of MPC. Because of this, some authors implicitly restrict the use of the term MPC to DMC, MAC/IDCOM, and their commercial variants. Other researcher include almost every long range predictive control (LRPC) algorithm in MPC category.

For the purpose of this thesis MPC is defined to be an optimal control strategy with following elements/characteristics:

- An explicit process model, for example impulse/step response, transfer function or state space. A combination of these model forms can also be considered.
- An explicit or implicit noise/disturbance model.

- An output/prediction horizon.
- An optimal long range output predictor based on the process and noise models.
- A user specified performance criterion with or without constraints on the input/output.
- Solution of an optimization problem to yield current and future control actions.
- Receding horizon implementation (i.e. implementing only the current control move and resolving the optimization problem at every sampling instant).

A number of modern process control strategies contain the above mentioned features and hence can be classified as MPC. Some of these include:

DMC	Dynamic Matrix Control (Cutler, 1979)
MAC / IDCOM	Model Algorithmic Control / Identification & Command (Richalet *et al.*, 1978)
GPC	Generalized Predictive Control (Clarke *et al.*, 1987a)
EHAC	Extended Horizon adaptive Control (Ydstie, 1985)
EPSAC	Extended Predictive Self Adaptive Control (De Keyser, 1985)
OCS	Optimum Control Synthesis (Peterka, 1982)
MUSMAR	Multistep Multivariable Adaptive Control (Greco *et al.*, 1984)
MURHAC	Multipredictor Receding Horizon Adaptive Control (Lemos and Mosca, 1985)

The above listed MPC's differ from each other with respect to process and noise model forms, performance criterion details, assumptions about future control moves and optimization problem solving methodologies. Three of these MPC's, DMC, MAC/IDCOM and GPC, warrant more detailed description because of their widespread popularity in industry and academia and are discussed in following sections.

1.1.1 Dynamic Matrix Control (DMC)

DMC algorithm was first developed at Shell and was introduced in the open literature in 1979 when Cutler reported successful application of this algorithm to a fluid catalytic cracking (FCC) unit. Some of the main features of DMC include the following:

- Step response process model
- An implicitly assumed Random Walk noise model.
- A quadratic cost function
- The notion of a control horizon (i.e. control moves after a certain point in future are arbitrarily set to zero).
- Systematic handling of process constraints through the use of a numerical optimization algorithm to calculate the required control action.

Some commercial versions of DMC include DMI (Dynamic Matrix Identification), a proprietary multivariable step-response identification software, as part of the MPC package.

Although DMC has been successfully employed in industry, it has some inherent drawbacks. Some of them are:

- A large number (typically >30) of step response coefficients are needed.
- It can not handle open-loop unstable processes.
- Disturbance rejection is exclusively dependent on servo response.

However, some versions of DMC use ad hoc fixes to go around some of the above listed problems.

1.1.2 Model Algorithmic Control / Identification and Command (MAC / IDCOM)

Model Algorithmic Control (MAC) developed by Richalet *et al.* (1978) is considered to be one of the two pioneering MPC schemes. The other is DMC which has been already discussed. Industrial implementation of MAC is also known as Identification and Command (IDCOM). The main elements of MAC/IDCOM are:

- Impulse response process model.
- Reference trajectory, a weighted sum of current and future output and current setpoint.
- Fixed output horizon, determined by the settling time.
- Control horizon set equal to the output horizon.
- Quadratic cost function.
- Receding horizon implementation.
- Two-step QP solution of the constrained optimization problem.

Although IDCOM uses an impulse response model it is recommended that first a transfer function model be identified which can subsequently be used to generate the required impulse response coefficients. This is due to the fact that the identification of impulse response from noisy input/output data results in coefficients with large variances.

The deficiencies of IDCOM are similar to those of DMC and include the following:

- A large number of impulse response coefficients are needed.
- Limited to open-loop stable processes.
- No separate tuning knob for disturbance rejection.

1.1.3 Generalized Predictive Control (GPC)

Unlike DMC and IDCOM the Generalized Predictive Control (GPC) was not a result of industrial ventures, rather it was a product of academic research at Oxford university. The landmark papers of Clarke *et al.* (1987a and 1987b) started a new era of systematic formulation of MPC. Key features of GPC are:

- Transfer function process model
- ARIMA noise model which includes the process model denominator.
- Recursive solution of Diophantine equations.

- Unifies many previously existing MPC schemes into one algorithm

Because of the equation error noise modeling, GPC can handle both open-loop stable and unstable processes. Since its introduction, GPC has enjoyed widespread academic acceptance. A large number of research papers have been devoted to alternate formulations, theoretical analysis and extension of the GPC philosophy. Industrial applications of GPC are also reported. However due to the lack of any dedicated marketing of a commercial version, it has not penetrated as deep into the industry as DMC or IDCOM. Some of the shortcomings of the GPC algorithm are:

- Noise dynamics are tied to the process dynamics.
- Output predictor lumps process and noise terms.
- Computationally expensive Diophantine equation solution.
- Relatively complicated control algorithm formulation compared to DMC.

1.1.4 Commercial MPC Packages

As has been mentioned above, the industrial implementation of MPC's preceded their formal formulation. This lead to the widespread acceptance and growing applications of MPC in today's industry. Hundreds of industrial applications of MPC, using different commercial packages, have been reported (Froisy, 1994). The following list summarizes some of the commercial packages for MPC (Ogunnaike and Ray, 1994):

DMC	Dynamic Matrix Control: marketed by DMC Corporation
QDMC	Quadratic DMC: developed and used by Shell
OPC	Optimum Predictive Control: developed and marketed by Treiber Controls Inc. and is almost identical to DMC except that the constraints are handled using linear programming rather than quadratic programming.
IDCOM	Identification and Command: an industrial implementation of MAC, is marketed by Setpoint Inc. with different names including IDCOM-S, IDCOM-M and IDCOM-B etc.. Originally IDCOM was developed in France by Adersa/Gerbios.

PC	Predictive Control: combines features of IDCOM and DMC and is marketed by Profimatics.
HPC	Horizon Predictive Control: a multi input single output (MISO) control scheme marketed by Honeywell Corporation. It differs from DMC in objective function and tuning parameters, but incorporates the concept of *correction horizon* beyond which the prediction error is to be reduced permanently to zero.
HMPC	Honeywell Multivariable Predictive Control: a multi input multi output (MIMO) control scheme marketed by Honeywell Corporation. It is a multivariable extension of HPC.

Most of the above listed commercial MPC algorithms have been implemented using a general purpose process control computer (e.g. a Digital VAX) which in turns communicates with a separate distributed control system (DCS) (e.g. a Honeywell TDC 3000 DCS system). Compact versions of MPC (e.g. OPC and IDCOM-B) may directly be implemented on DCS

A brief list of industrial applications of MPC includes control of the following (Ogunnaike and Ray, 1994):

- Distillation columns
- Fluid catalytic cracking (FCC)
- Hydrocracker reactor
- Polymerization
- Chemical reactors
- Pulp and paper making
- Polymer extruders
- Tar sand extraction cells

1.2 Advantages of MPC

It can be rightfully asserted that at present MPC is the most popular advanced process control strategy in the petroleum and petrochemical industry. The number of industrial applications of MPC is growing rapidly because of its inherent advantages over conventional PID controllers. Another reason is the easy availability of MPC technology at affordable prices and the ever falling computing costs. A number of advantages of a standard MPC package are listed below:

- Can handle Multi-Input-Multi-Output (MIMO) systems
- Easy to deal with time delays in a systematic way
- Incorporates process input/output constraints
- Very flexible tuning parameters/strategies
- Handles unusual process dynamics, e.g. "wrong-way" initial responses.
- Allows for on-line real-time-optimization (RTO)
- Fits well with model-based process monitoring
- Intuitively-appealing, optimization-based control
- Systematic compensation for noise/disturbances
- Proven performance in the chemical industry

1.3 Drawbacks of MPC

Naturally, in this imperfect world, advantages are offset by shortcomings or drawbacks. Some of the deficiencies of MPC are:

- Computational complexity.
- Tuning parameters are not in terms of usual performance specification criteria.
- Specification of noise/disturbance models is not well defined.
- Robustness issues are not explicitly addressed.
- Theoretical analysis is difficult for finite horizon controllers.

1.4 Structure of the Thesis

This section consists of two sub-sections. Sub-section-1.4.1 describes the motivation for the study of specific aspects of MPC and establishes the objectives/goals for the research work. The second sub-section describes the organization of the thesis.

1.4.1 Motivation and Objectives

The rapidly growing number of MPC industrial applications presents a serious challenge to academia in terms of developing a more fundamental understanding and theoretical analysis of MPC. One important area of interest is to integrate the seemingly different MPC schemes. While different applications may be situation-specific, unification, in terms of the functional characteristics of MPC algorithms is highly desirable. It is not important that the unification should result in a single generalized master algorithm, all that is needed is a conceptual unification which will bring the currently scattered MPC strategies under a common framework so that parameters in various schemes can be mutually translatable. Some of the bases for unification include the following:

- Process model type: transfer function, step, impulse or state space.
- Noise/disturbance modeling: equation error or output error etc..
- Reference trajectory: reference time in IDCOM and model-following in GPC etc..
- Cost function
- Control law structure

Complete unification covering all the characteristics of MPC's is a very broad research area and is too unrealistic to attempt by an individual or even by a single research group. However, *unification on the basis of one or two criteria* seems practical and has been successfully attempted in the past. Based on different horizon settings GPC unifies many previously developed control strategies. Li *et al.* (1989) state space formulation of MPC started an era of process model based unification. Qi and Fisher (1993) developed a dual-model MPC which uses a combination of step response and transfer function process models. Soeterboek (1990 and 1992) and Soeterboek *et al.* (1990) used a generalized equation error noise/disturbance model in their Unified Predictive Control (UPC) algorithm. The work presented in this thesis employs *a more general noise/disturbance model* which includes both the output error and the equation error structures. A very important issue in process control applications is the disturbance rejection properties of the implemented control system. Most of the existing MPC schemes do not deal with this point explicitly. For example, DMC has no servo-independent tuning mechanism to control the speed of disturbance rejection. GPC assumes that disturbances have the same dynamics as the process. In general there has been no significant emphasis placed on the role of noise/disturbance model in disturbance rejection performance of MPC. In this thesis, *a general noise/disturbance model is introduced and used as a tuning mechanism that can be utilized to achieve desired disturbance rejection properties.*

Transfer function based MPC's employ Diophantine equations to separate the free and forced (past and future) responses. Usually the process and noise models are lumped in the derivation of the optimal output predictor. This lumped formulation restricts the independent manipulation of the noise model. It also complicates the analysis of the resulting controller. A separated Diophantine predictor formulation of MPC is proposed in the current work. It leads to an MPC algorithm that is simple, straightforward and allows separate manipulation of the noise/disturbance model.

A great deal of computational load of an MPC algorithm is associated with the Diophantine equations that appear in output predictors even though the future outputs are not explicitly calculated. In addition to the computational load, the number of Diophantine equations is proportional to the output horizon and pose an additional burden on computer storage. The computational load and the storage requirements become a serious limitation in MIMO systems. The search for *computationally cheaper MPC algorithms* is another objective of the thesis.

Most existing MPC strategies require large output horizons in order to yield robust control in the presence of the inevitable unmodeled dynamics. A goal of the current research is to *generalize the performance criterion (cost function)* of MPC so that it can use a sparse output horizon. Another limitation of currently available MPC packages is the integer nature of the control horizon. Interpolation of the control horizon is considered as another objective for the study.

The concept of long range prediction in MPC can be extended to include the steady state prediction in the cost function. This idea has already been implemented by Kwok and Shah (1994). However in this thesis *the steady state prediction* has been incorporated in MPC using an entirely new interpretation and methodology.

A recursive MPC formulation, which gives the control law for a control horizon of N_u+m in terms of the control for N_u, is also on the list of objectives of the current study.

1.4.2 Thesis Organization

The technical material of the thesis begins with *Chapter-2* which discusses the structure and role of noise/disturbance models in detail. A *generalized noise/disturbance model*, which covers both the equation and output error structures, is developed for inclusion in the UMPC formulation. The chapter also describes a *modified form of the classical DMC algorithm* which includes an independent tuning parameter for disturbance rejection. In the same chapter, *an overparameterized optimal output predictor* is developed to reduce computational load and computer storage requirement.

The *Separated Diophantine Predictor (SDP)* is developed in *Chapter-3*. Using this predictor a modified form of the classical DMC is formulated, which includes parameters for independent disturbance rejection and noise attenuation,. A new concept, that of *disturbance horizon*, is introduced. Strategies to incorporate more than one noise/disturbance model into the MPC formulation has been included using the notion of disturbance horizon and *σ-weighting*. Detailed analyses of SDP are included to relate it to the classical predictors. An *overparameterized SDP* is proposed to reduce the computer time and storage requirements.

Chapter-4 of the thesis deals with the development of computationally more efficient *Reduced Diophantine Predictors (RDP)*. A parsimonious and an overparameterized form are formulated and then mathematically proven to be identical to the classical optimal predictors. The sources of computational saving are discussed in detail.

The notion of *a sparse output horizon* is introduced using a generalized cost function in *Chapter-5*. The advantages of using an integral cost function for highly sparse output horizons is demonstrated and a more general control policy is introduced. Another new concept, *control horizon interpolation*, using a sparse control horizon is included. This chapter also presents a method for using *multiple control horizons* in the UMPC formulation.

Chapter-6 is dedicated to the *steady state error weighting formulation* of UMPC. A completely new basis for including the steady state tracking error in MPC is introduced. Two methods for steady state output prediction are presented. The steady state error weighting formulation of MPC is shown to be a weighted interpolation of dynamic and mean-level MPC. The steady state error weighting is *normalized* and the concept of *disturbance horizon*, developed in *Chapter-3*, is applied to this scenario.

A *control horizon-recursive formulation* is put forward in *Chapter-7*. The matrix inversion lemma is used to avoid the matrix inversion normally involved in the control calculations. Two new methods of *control horizon interpolation* (one was developed in *Chapter-4*) are presented. A method of simultaneous recursion and interpolation of control horizon is also developed.

The presentation of the current research work is *concluded in Chapter-8* of the thesis. This chapter draws overall conclusions about the accomplished research results presented in chapters 2-7. Guidelines for potential future research work in the area of model predictive control are provided at the end of this last chapter.

References:

Clarke, D.W., Mohtadi, C. and Tuffs, P.S., " Generalized Predictive Control - Part I the Basic Algorithm ", Automatica, Vol. 23, No. 2, pp.137-148, 1987a.

Clarke, D.W., Mohtadi, C. and Tuffs, P.S., " Generalized Predictive Control - Part II Extensions and Interpretations ", Automatica, Vol. 23, No. 2, pp. 149-160, 1987b.

Clarke, D.W.," Adaptive Generalized Predictive Control ", Proceedings of the Fourth International Conference on Process Control (CPCIV), Editors Y. Arkun and W. H. Ray, Padre Island, TX, 1991.

Cutler, C.R., "Dynamic Matrix Control - A Computer Control Algorithm ", AIChE National Meeting, Houston, TX, 1979.

De Keyser, M.C., "A Comparative Study of Self-Adaptive Long Range Predictive Control Methods", Proceedings of the 7th IFAC Symposium on Identification and System Parameter estimation, pp. 1317-1322, 1985.

De Keyser, M.C., "Basic Principles of Model Based Predictive Control", Proceedings of the First European Control Conference, pp. 1753-1758, 1991.

Froisy B.J., "Model Predictive Control: Past, Present and Future", ISA Transactions, Vol. 33, No. 3, pp. 235, 1994.

Greco, C., "Performance Improvement of Self-tuning Controllers by Multistep Horizons: The MUSMAR Approach", Automatica, Vol. 20, pp. 681-699, 1984.

Kwok, K. and Shah, S. L., "Long Range Predictive Control with a Terminal Matching Condition", Chemical Engineering Science, Vol. 49, No. 9, pp. 1287-1300, 1994.

Lemos, J.M. and Mosca, E., "A Multipredictor-based LQ Self-tuning ", Proceedings of the 7th IFAC Symposium on Identification and System Parameter estimation, pp. 137-141, 1985.

Li, S., Lim, K.L. and Fisher, D.G., "A State Space Formulation for Model Predictive Control ", A.I.Ch.E. Journal, Vol. 35, No. 2, pp. 241-249, 1989.

Muske, K.R. and Rawlings, J.B. "Model Predictive Control with Linear Models", AIChE Journal, Vol. 39, No. 2, 1993.

Ogunnaike, B.A. and Ray, W.H., "Process Dynamics, Modeling, and Control", Oxford University Press, New York, 1994.

Peterka, V., "Predictor Based Self-tuning Control", Proceedings of the 6th IFAC Symposium on Identification and System Parameter estimation, pp. 873-878, 1982.

Qi, Kent Z. and Fisher, D.G., "Model Predictive Control for Open-Loop Unstable Processes", Proc. American Control Conference, San Francisco, CA, pp. 796-800, 1993.

Richalet, J., Rault, A., Testud, J.L., and Papon, J., "Model Predictive Heuristic Control: Applications to Industrial Processes", Automatica, Vol. 14, pp. 413, 1978.

Richalet, J. A., "Observations on Model-Based Predictive Control", Control Engineering, Vol. 39, No. 10, pp. 39, 1992.

Soeterboek, A.R.M., "Predictive Control — A Unified Approach", Ph.D. thesis, Delft University of Technology, Netherlands, 1990.

Soeterboek, A.R.M., "Predictive Control — A Unified Approach", Prentice Hall International Series in Systems and Control Engineering, 1992.

Soeterboek, A.R.M., Verbruggen, H.B., Bosch, P.P.J., and Butler,H., "On the Unification of Predictive Control Algorithms", Proceedings of the 29th Conference on Decision and Control, Honolulu, Hawaii, 1990.

Ydstie, B. E., "Theory and Application of the Extended Horizon Self-tuning Controller", AIChE Journal Vol. 31, pp. 1771-1780, 1985.

Chapter 2

Unified Model Predictive Control [1]

2.1 Introduction

A model-structure-unified formulation of long-range predictive control is presented in this chapter. This model-based control strategy which shall in the sequel be called Unified Model Predictive Control (UMPC), uses a generalized polynomial structure for both process and noise models rather than a specific model structure like ARIMAX. A wide range of model structures can be handled and by selecting the appropriate model structure, the Unified Model Predictive Control law may be reduced to well known controllers like GPC and DMC.

The role of the noise model as a tuning parameter for the speed of disturbance rejection is elaborated and the advantages of a generalized noise model are established. A unified formulation is adopted for the two complementary classes of noise model structures viz. the Equation Error (EE) model and the Output Error (OE) model.

UMPC is derived using two different predictors. The Lumped Diophantine Predictor (LDP) is an extension of the classical long-range predictor used in controllers such as GPC but incorporates both the EE and the OE noise model structures in one formulation. A Lumped Diophantine Overparameterized Predictor (LDOP) was then developed which offers the same generality but with a substantial saving in computational load. The UMPC is a receding horizon controller and retains the concepts of output horizon and control horizon and control weighting (λ) used in GPC and other modern controllers.

---

[1] An earlier version of this chapter was presented at the 44th Canadian Chemical Engineering Conference : "Unified Predictive Control", Saudagar, M.A., Fisher, D.G. and Shah, S.L., 1994.

An improved version of classical DMC, IDMC, is derived, as a special case of the generalized noise structure, using the LDP. The IDMC algorithm, which adds a first order lag filter in the DMC noise model, provides independent tuning for disturbance rejection.

A number of simulation results are presented to illustrate the effect of various noise model structures and the performance of the newly formulated IDMC algorithm.

One of the objectives of the present work is to integrate all such structure-specific linear controllers into a single algorithm using a general model-structure.

2.2 The Generalized Model Structure

The measured output can be described by the following equation:

Output Signal = Process Model × Input Signal + Noise Model × Uncertainty Signal

or

$$y(t) = G(q^{-1})u(t-1) + W(q^{-1})e(t) \tag{2.1}$$

where $G(q^{-1})$ is the process model, $W(q^{-1})$ is the noise model and $y(t)$, $u(t)$ and $e(t)$ are the output, the input and the uncertainty signals at time t respectively. A typical example of the uncertainty signal $e(t)$ is zero-mean, normally-distributed, independent, white measurement noise.

The process model $G(q^{-1})$ can be a non-parameteric finite *impulse*-response model as employed in Identification and Command (IDCOM), (Richalet et al., 1978), a non-parameteric *step*-response model such as the one used in Dynamic Matrix Control, (DMC), (Cutler, 1979) or it can be a parameteric *transfer function* model as employed in Generalized Predictive Control (GPC). Moreover these model forms, parameteric and non-parameteric, can also be rearranged into the more descriptive state space formulations (Lim, 1988, Li et al., 1989 and Lundstrom et al., 1991).

In UMPC the following discrete time transfer function type process model is used,

$$G(q^{-1}) = \frac{B(q^{-1})}{A(q^{-1})} \tag{2.2}$$

where $A(q^{-1})$ and $B(q^{-1})$ are polynomials in the backward shift operator q^{-1} which will be omitted from all polynomials for simplicity of notation in the rest of the chapter.

The primary purpose of including a noise model in the measured output equation is to inform the controller about the dynamic structure of the noise/disturbance. Because of the stochastic nature of the noise and the disturbance, detailed identification of the noise model W is usually not feasible. In practice the most useful specification of the noise model has been the inclusion of an integrator which for the discrete-time case is given as:

$$\text{Integrator} = \frac{1}{\Delta} = \frac{1}{1-q^{-1}} \qquad \text{with} \qquad \Delta = 1 - q^{-1} = \text{Difference Operator} \tag{2.3}$$

An integrator in the noise model successfully describes random step disturbances introduced at the output and the drifting integrated white noise associated with output measuring devices. The steady-state error due to a difference between the gain of the actual process and the process model is also well-described by this integrator. Most of the recent LRPC's include an integrator in their noise models. Besides the integrator, the remainder of the noise model is either specified on an ad hoc basis or is left unspecified. The noise models of DMC and GPC are:

$$W = \frac{1}{\Delta} \qquad \text{DMC Noise Model} \tag{2.4}$$

$$W = \frac{C}{A\Delta} \qquad \text{GPC Noise Model} \tag{2.5}$$

The DMC noise model consists of only a single integrator, while the GPC noise model contains the process model denominator A in addition to an integrator and an observer polynomial C. The inclusion of A in the noise model is appropriate when the disturbances and the plant have the same dynamics. However in general this "hard-wired" A reduces the flexibility in specifying the noise model, and in many cases it amplifies the effect of the unmodelled dynamics. Moreover the A in the noise model can not describe measurement noise which is almost always added at the output i.e. it does not share the plant dynamics.

In controller design a rather important role of the noise model is its influence on the speed of disturbance rejection. Although this feature of the noise model has been implicitly utilized in some of the LRPC's, it has not been emphasized explicitly in the LRPC literature. As the main issue in process control is regulatory control, i.e. keeping specified process variables at their desired values (set-points) in the presence of internal and external disturbances and noise, the speed of disturbance rejection is of prime importance. In this context a generalized noise model structure provides more flexibility in controlling disturbance rejection than using an arbitrarily fixed structure.

As mentioned earlier, the long range predictive controllers developed to date use arbitrarily specified noise models. A general noise model structure for discrete-time linear time invariant systems has been widely used in the time series literature (Box and Jenkins, 1976). Harris and MacGregor (1986) employed such structures to develop Linear Quadratic (LQ) controllers. Soeterboek (1990 and 1992) and Soeterboek *et al.* (1990) used a generalized equation error (ARARMAX) noise/disturbance model in their Unified Predictive Control (UPC) algorithm. De Keyser (1991) suggested the use of a Box-Jenkins type model structure for LRPC. However no LRPC based on these generalized noise model structures has gained any significant acceptance.

The following discrete-time rational polynomial model structure, based on (2.1), is considered to be the most general model structure for linear rational polynomial noise models (Ljung, 1987 and Soderstrom, 1989):

$$W = \frac{C}{D'} \tag{2.6}$$

where C and D' are polynomials in the backward shift operator q^{-1}

For controller design an integrator is generally included in the noise model for offset removal. For example GPC and DMC use a single integrator $1/\Delta$ ($\Delta=1-q^{-1}$) in the noise model. However offset removal may require more than one integrator for ramp and higher order disturbances or set points. Therefore a multiple integrator $1/\Delta^n$ was included in the UMPC model structure to offer a more general formulation.

Many popular model structures including ARMAX and ARIMAX use the process denominator polynomial A in the denominator of the noise model. These model structures are called Equation-Error (EE) models. The model structures which do not contain the process denominator polynomial A in the denominator of the noise model are called Output Error (OE) models. In general controllers based on the EE model structure result in faster disturbance rejection than those based on the OE structure. For example GPC is much faster than DMC in terms of disturbance rejection. However in the presence of higher Model Plant Mismatch (MPM) these EE model based controllers become unstable due to the presence of the process denominator polynomial A in the denominator of the noise model. One indirect solution, used in GPC, is to use a compensating polynomial (the so-called T-filter or observer polynomial) in the numerator of the noise model. In UMPC, in addition to the C polynomial in the numerator, an arbitrary polynomial, D, is included in the denominator of the noise model rather than restricting the denominator to the fixed process denominator polynomial A.

The D polynomial provides a systematic way of increasing the speed of disturbance rejection. The D = 1 case gives a DMC-type controller which has slow disturbance rejection speed. With a $D=1-\alpha q^{-1}$, increasing α increases the speed of disturbance rejection. Setting $D = A$ gives the GPC disturbance rejection characteristic. In order to include the optional process polynomial A, the arbitrary polynomial D and the multiple integrator in the UMPC formulation, the denominator polynomial of the noise model (2.1) is defined as follows:

$$\text{Noise model denominator} = D' = D\breve{A}\Delta^{n} \tag{2.7}$$

and the noise model for (2.1) becomes:

$$W = \frac{C}{D\breve{A}\Delta^{n}} \tag{2.8}$$

The notation $\breve{A}$ is used to indicate that the polynomial A is included for equation error structures. More specifically:

$\breve{A} = A$	for EE structures
$\breve{A} = 1$	for OE structures

The $\breve{A}$ notation is an extra burden but it unifies the formulations for the EE and the OE structures into a single formulation. Without this notation two parallel derivations would be needed to encompass both the EE and the OE noise structures. Note that although the optional polynomial A could be included in D, it is shown explicitly in the denominator of the noise model in (2.9) as a separate term for the following two reasons:

- It explicitly highlights the EE model structure option.
- In the case of the EE model structure, A gets cancelled out during the controller derivation resulting in a simpler formulation. In other words including A as a separate term in the denominator of the noise model assures the relative primeness of the polynomials A and D.

With the above definition of the noise model and the transfer function process model, the following generalized model structure for the measured output is used in UMPC:

$$y(t) = \frac{B}{A}u(t-1) + \frac{C}{D\breve{A}\Delta^{n}}e(t) \tag{2.9}$$

UMPC Model

In cases when only the effect of noise, disturbances and/or model uncertainty on the process output is of interest the following model is more appropriate:

$$y(t) = \frac{B}{A}u(t-1) + x(t) \tag{2.10}$$

where $x(t)$ is the residual at time t.

Setting $D = 1$, $n = 1$ and $\breve{A} = A$ in the generalized noise/disturbance model (2.9) gives the following ARIMAX (or CARIMA) model used in GPC:

$$Ay(t) = Bu(t-1) + \frac{C}{\Delta}e(t) \tag{2.11}$$

For model-following control, optional polynomials Θ and Ω may be used in the general model of (2.9) to give:

$$\frac{\Theta}{\Omega}y(t) = \frac{B}{A}u(t-1) + \frac{C}{D\breve{A}\Delta^{n}}e(t) \tag{2.12}$$

However in many cases $\Theta = \Omega = 1$ and even if they are arbitrary polynomials, they can always be merged into A, B, C, D.

It may be noted that (2.12) can be rearranged into the form:

$$\left[\Theta A D \Delta^{n-1}\right]y(t) = \left[B \Omega D \Delta^{n-1}\right]u(t-1) + \frac{\left[C\Omega\hat{A}\right]}{\Delta}e(t) \tag{2.13}$$

Where $\hat{A}$ is defined as follows:

$\hat{A} = 1$ for EE structures

$\hat{A} = A$ for OE structures

and

$$\hat{A}\breve{A} = A$$

By redefining the coefficient polynomials (2.13) can be written as:

$$A^{\bullet}y(t) = B^{\bullet}u(t-1) + \frac{C^{\bullet}}{\Delta}e(t) \tag{2.14}$$

which has the same structure as the model (2.11) used in GPC. However, as is obvious from the above and as will be discussed later, this structure lumps the *process* characteristics (A, B) with the *noise* characteristics (C, D, Δ). One of the objectives of this thesis is to separate the noise versus process characteristics and to design a controller which allows independent specification/tuning of the noise/disturbance rejection properties. Therefore the more general *separated* model structure given in (2.9) is used in UMPC. Note that the simplification, in the model following case, by merging Θ and Ω, into *A*, *B*, *C* and *D* does not disturb the relative contributions of process and noise terms model terms in the prediction and is therefore not contradictory to the philosophy of separate parameterization.

2.3 The Lumped Diophantine Predictor (LDP)

For *j*-step ahead predictions the time parameter in equation (2.9) is shifted by *+j* giving:

$$y(t+j) = \frac{B}{A}u(t+j-1) + \left[\frac{C}{D\breve{A}\Delta^n}\right]e(t+j) \tag{2.15}$$

Note that $y(t+j)$ is the future output and is not available at current time t . The noise model $\left[\frac{C}{D\breve{A}\Delta^n}\right]e(t+j)$ contains both future and past/present terms in $e(.)$. The past and present values of $e(.)$ can easily be reconstructed using (2.9), while the future value of $e(.)$ can at best be set equal to its expected value which for the case of zero mean white noise is equal to zero. However to do this, the noise term must be divided into future and past/present components. Usually this segregation is done using the well-known Diophantine equations.

$$\left[\frac{C}{D\breve{A}\Delta^n}\right] = E_j + q^{-j}\frac{F_j}{D\breve{A}\Delta^n} \tag{2.16}$$

In the classical approach for obtaining the *j*-step ahead predictor (used in GPC for example), the noise model Diophantine expansion (2.16) is also employed when the process model (first term on the RHS of equation (2.15)) is divided into past/future terms. This results in the coupling of the process model and noise model polynomials. In the present study the predictor based on this classical approach is referred to as Lumped Diophantine Predictor or LDP because it "lumps" the process and noise polynomials together. The derivation of the LDP is as follows.

The first term on the RHS of equation (2.15) is rearranged as follows:

$$y(t+j) = \left[\frac{C}{D\breve{A}\Delta^n}\right] DB \frac{\Delta^n u(t+j-1)}{C\hat{A}} + \left[\frac{C}{D\breve{A}\Delta^n}\right] e(t+j) \tag{2.17}$$

Substituting (2.16) for $\left[\frac{C}{D\breve{A}\Delta^n}\right]$ in (2.17) gives:

$$y(t+j) = \left[E_j + q^{-j}\frac{F_j}{D\breve{A}\Delta^n}\right] DB \frac{\Delta^n u(t+j-1)}{C\hat{A}} + \left[E_j + q^{-j}\frac{F_j}{D\breve{A}\Delta^n}\right] e(t+j) \tag{2.18}$$

The j-step ahead optimal prediction conditioned on the input/output data up to time t is denoted by $y_p(t+j|t)$ and is obtained by setting the future component $E_j e(t+j)$ equal to zero and rearranging (2.18) to:

$$y_p(t+j|t) = \frac{E_j DB}{C\hat{A}} \Delta^n u(t+j-1) + \frac{F_j}{C}\left[\frac{B}{A}u(t-1) + \frac{C}{D\breve{A}\Delta^n}e(t)\right] \tag{2.19}$$

Then recognizing from (2.9) that $\left[\frac{B}{A}u(t-1) + \frac{C}{D\breve{A}\Delta^n}e(t)\right] = y(t)$ the above equation can be simplified as follows:

$$y_p(t+j|t) = \left[\frac{E_j DB}{C\hat{A}}\right] \Delta^n u(t+j-1) + \frac{F_j}{C} y(t) \tag{2.20}$$

This eliminates the unknown future $e(t+j)$ terms from the output predictor but the first term on the RHS of (2.20) contains both, future and past components. In order to separate these components the following Diophantine equation is used:

$$\left[\frac{E_j DB}{C\hat{A}}\right] = \tilde{G}_j + q^{-j}\frac{\overline{G}_j}{C\hat{A}} \tag{2.21}$$

Substitution of (2.21) in (2.20) leads to

$$y_p(t+j|t) = \tilde{G}_j \Delta^n u(t+j-1) + \frac{\overline{G}_j}{C\hat{A}} \Delta^n u(t-1) + \frac{F_j}{C} y(t) \tag{2.22}$$

With the following definitions of filtered input and output

$$\Delta^n u^f(t-1) = \frac{\Delta^n u(t-1)}{C\hat{A}} \quad \text{and} \quad y^f(t) = \frac{y(t)}{C} \tag{2.23}$$

the predictor (2.22) becomes:

$$\boxed{y_p(t+j|t)=\widetilde{G}_j\Delta^n u(t+j-1)+\overline{G}_j\Delta^n u^f(t-1)+F_j y^f(t)} \tag{2.24}$$

LDP

Remark 2.3-1:

The predictor (2.24) is termed the lumped Diophantine predictor or LDP because the second term on the RHS is a combination of process and model polynomials.

Remark 2.3-2:

The first term on the RHS of (2.24) is the *forced response* which depends on the present and *future* input values, while the remaining two terms on the RHS constitute the prediction of the *free response* of the system based strictly on input-output information until time = t, i.e. when $\{u(t+j-1),\ j=1,2,3,\ldots\}=0$.

Remark 2.3-3:

It can be shown that the forced response in the LDP is independent of the noise model. Equation (2.21) gives

$$\widetilde{G}_j=\left[\frac{E_j DB}{C\bar{A}}\right]_j \tag{2.25}$$

where the square brackets with subscript j denotes the first j terms of the argument within the square brackets. Moreover

$$E_j=\left[\frac{C}{D\breve{A}\Delta^n}\right]_j \tag{2.26}$$

Substituting (2.26) in (2.25) gives:

$$\widetilde{G}_j=\left[\frac{C}{D\breve{A}\Delta^n}\frac{DB}{C\bar{A}}\right]_j=\left[\frac{B}{\breve{A}\Delta^n}\right]_j \tag{2.27}$$

Equation (2.27) shows that $\widetilde{G}_j$ depends only on process model polynomials (B and A) which in turn proves that the forced response does not depend on the noise model.

The second term on the RHS of the LDP contains filters which depend on both process and noise models, while the third term depends only on the specified noise model.

Remark 2.3-4:

The LDP may be derived in many other ways, however the foregoing derivation was developed to specifically highlight the fact that the *process* transfer function is expanded into future and past parts using the *noise* model Diophantine equation (see equations (2.17) and (2.18)).

Remark 2.3-5:

Assuming that (2.9) accurately represents the actual process and assuming perfect process and noise modelling, the prediction error, $\varepsilon_p(t+j|t)$, of the LDP corresponding to $y_p(t+j|t)$, can easily be obtained by subtracting (2.19) from (2.18) as follows:

$$\varepsilon_p(t+j|t) = y(t+j) - y_p(t+j|t) = E_j e(t+j) \tag{2.28}$$

This is also obvious from the fact that the assumption made in deriving (2.19) from (2.18) was that $E_j e(t+j) = 0$.

Remark 2.3-6:

For ARIMAX (or CARIMA) models $\breve{A} = A$ and the LDP gives the GPC predictor with a single integrator (i.e. n=1), while for DMC-like controllers with $C = D = \breve{A} = 1$ the noise model is simply a single integrator, and therefore $F_j = 1$ (and the I/O filtering in (2.23) is not required).

Remark 2.3-7:

For OE noise model structures the second term on the RHS of equation (2.22) contains the denominator of the process transfer function. This implies that for open loop unstable processes the predictions may become unbounded. A simple way to get stable predictions for unstable processes is to use an EE noise model structure instead of an OE structure.

Another option revealed by the above derivation of the LDP is to use a partial EE noise model structure which is defined as one containing only the unstable factors of the process model transfer function denominator instead of the whole denominator.

$$y(t) = \frac{B}{A}u(t-1) + \frac{C}{DA_u\Delta^n}e(t) \qquad \text{where} \quad A_u = \text{ unstable factors of } A \tag{2.29}$$

$$A = A_s A_u \qquad \text{where} \quad A_s = \text{ stable factors of } A \tag{2.30}$$

$$\frac{C}{DA_s\Delta^n} = E_j + q^{-j}\frac{F_j}{DA_s\Delta^n} \tag{2.31}$$

The same effect (i.e. the partial EE modelling) may also be achieved by simply setting one of the factors of C equal to A_s.

A block diagram illustrating the LDP is given in Figure 2.1. Note that the predictor does not give separate free responses due to the manipulated variable and uncertainty. In Figure 2.2 the free response for the LDP is shown to be a sum of the free responses due to the past manipulated variable and uncertainty.

2.4 The Lumped Diophantine Overparameterized Predictor (LDOP)

The past and future terms in the noise model can be separated using the following Diophantine expansion rather than (2.16):

$$\frac{C}{D\breve{A}\Delta^n} = E_N + q^{-N}\frac{F_N}{D\breve{A}\Delta^n} \tag{2.32}$$

where N is any integer greater than or equal to j. A useful case results when N is set equal to the maximum output horizon of a long range predictive controller.

Substituting (2.32) for $\frac{C}{D\breve{A}\Delta^n}$ in (2.17) gives:

$$y(t+j) = \left[E_N + q^{-N}\frac{F_N}{D\breve{A}\Delta^n}\right]DB\frac{\Delta^n u(t+j-1)}{C\hat{A}} + \left[E_N + q^{-N}\frac{F_N}{D\breve{A}\Delta^n}\right]e(t+j) \tag{2.33}$$

$$y(t+j) = \frac{E_N DB}{C\hat{A}}\Delta^n u(t+j-1) + \frac{F_N}{C}\left[\frac{B}{\cdot A}u(t-N+j-1) + \frac{C}{D\breve{A}\Delta^n}e(t-N+j)\right] + E_N e(t+j) \tag{2.34}$$

The term in the square bracket on the RHS of the above equation is $y(t\text{-}N\text{+}j)$ and the expected future values of $e(.)$ are zero. Substituting these gives:

$$y_p(t+j|t) = \frac{E_N DB}{C\hat{A}}\Delta^n u(t+j-1) + \frac{F_N}{C}y(t-N+j) + \overline{E}_{Nj}e(t) \tag{2.35}$$

where

$$E_N = \tilde{E}_{Nj} + q^{-j}\bar{E}_{Nj} \tag{2.36}$$

Note that the future stochastic part $\tilde{E}_{Nj}e(t+j)$ is set to zero in arriving at equation (2.35)

Expanding the input into past and present/future terms using

$$\frac{E_N DB}{C\hat{A}} = \tilde{G}_N + q^{-N}\frac{\bar{G}_N}{C\hat{A}} \tag{2.37}$$

$$\tilde{G}_N = \tilde{\tilde{G}}_{Nj} + q^{-j}\bar{\tilde{G}}_{Nj} \tag{2.38}$$

gives

$$\begin{aligned} y_p(t+j|t) = \tilde{\tilde{G}}_{Nj}\Delta^n u(t+j-1) + \bar{\tilde{G}}_{Nj}\Delta^n u(t-1) + \bar{G}_N \frac{\Delta^n u(t-N+j-1)}{C\hat{A}} \\ + \frac{F_N}{C} y(t-N+j) + \bar{E}_{Nj}e(t) \end{aligned} \tag{2.39}$$

Defining $e(t)$ (using (2.9)) as

$$e(t) = \frac{DA\Delta^n y(t) - DB\Delta^n u(t-1)}{C\hat{A}} = D\breve{A}\Delta^n y^f(t) - DB\Delta^n u^f(t-1) \tag{2.40}$$

and the filtered input/output, $\Delta^n u^f$ and y^f as in (2.23) yields

$$\boxed{\begin{aligned} y_p(t+j|t) = \tilde{\tilde{G}}_{Nj}\Delta^n u(t+j-1) + \bar{\tilde{G}}_{Nj}\Delta^n u(t-1) + \bar{G}_N\Delta^n u^f(t-N+j-1) \\ + F_N y^f(t-N+j) + \bar{E}_{Nj}e(t) \end{aligned} \quad (2.41) \quad \text{LDOP}}$$

Remark 2.4-1:

The predictor (2.41) is termed as the lumped Diophantine overparameterized predictor or LDOP because the Diophantine expansions corresponding to the farthest horizon point is used for all *j*'s. The future polynomials of these Diophantine equations have *N* parameters and thus are overparameterized compared to the Diophantine equations corresponding to *j*th horizon point which contain only *j* parameters in their future polynomials.

Remark 2.4-2:

The first term on the RHS of LDOP is the *forced response* which depends on the present and future input values, while the remaining four terms on the RHS constitute the prediction of the *free response* of the system based strictly on input-output information until time = t.

Remark 2.4-3:

The computational load associated with equations (2.36) and (2.38) is trivial because $\widetilde{E}_{Nj}$ is simply the first j terms of E_N and $q^{-j}\overline{E}_{Nj}$ represents the last N-j terms of E_N. Similarly $\tilde{\tilde{G}}_{Nj}$ is the first j terms of $\tilde{G}_N$ while $q^{-j}\overline{\tilde{G}}_{Nj}$ represents the last N-j terms of $\tilde{G}_N$.

Remark 2.4-4:

The computational load of LDOP is much lower than that for LDP. This is primarily due to the fact that the two sets of Diophantine equations (each consisting of N_2 equations where N_2 = output prediction horizon) are replaced by only two Diophantine equations. The storage requirement is also reduced.

Remark 2.4-5:

LDOP is mathematically identical to LDP. The following theorem proves this.

Theorem-1:

The LDOP (equation 2.41)) is mathematically identical to the LDP (equation (2.24)).

Proof:

Combining (2.32) and (2.36) gives

$$\frac{C}{D\breve{A}\Delta^n} = \widetilde{E}_{Nj} + q^{-j}\overline{E}_{Nj} + q^{-N}\frac{F_N}{D\breve{A}\Delta^n} = \widetilde{E}_{Nj} + q^{-j}\frac{D\breve{A}\Delta^n\overline{E}_{Nj} + q^{-N+j}F_N}{D\breve{A}\Delta^n}$$

which implies that

$$E_j = \widetilde{E}_{Nj} \tag{2.42}$$

and

$$F_j = D\breve{A}\Delta^n\overline{E}_{Nj} + q^{-N+j}F_N \tag{2.43}$$

Substituting (2.38) in (2.37) and rearranging yields:

$$\frac{E_N DB}{C\hat{A}} = \tilde{\tilde{G}}_{Nj} + q^{-j}\overline{\tilde{G}}_{Nj} + q^{-N}\frac{\overline{G}_N}{C\hat{A}} = \tilde{\tilde{G}}_{Nj} + q^{-j}\frac{\overline{\tilde{G}}_{Nj}C\hat{A} + q^{-N+j}\overline{G}_N}{C\hat{A}} \tag{2.44}$$

Using (2.36), $\frac{E_N DB}{C\hat{A}}$ can also be given as:

$$\frac{E_N DB}{C\hat{A}} = \frac{E_j DB}{C\hat{A}} + q^{-j}\frac{\overline{E}_{Nj}DB}{C\hat{A}} = \tilde{G}_j + q^{-j}\frac{\overline{G}_j}{C\hat{A}} + q^{-j}\frac{\overline{E}_{Nj}DB}{C\hat{A}} \tag{2.45}$$

or

$$\frac{E_N DB}{C\hat{A}} = \tilde{G}_j + q^{-j}\frac{\overline{G}_j + \overline{E}_{Nj}DB}{C\hat{A}} \tag{2.46}$$

Comparing (2.44) and (2.46) gives:

$$\tilde{G}_j = \tilde{\tilde{G}}_{Nj} \tag{2.47}$$

$$\overline{G}_j + \overline{E}_{Nj}DB = \overline{\overline{G}}_{Nj}C\hat{A} + q^{-N+j}\overline{G}_N \tag{2.48}$$

or

$$\overline{G}_j = \overline{\overline{G}}_{Nj}C\hat{A} + q^{-N+j}\overline{G}_N - \overline{E}_{Nj}DB \tag{2.49}$$

Substituting $\tilde{G}_j$, $\overline{G}_j$ and F_j from equation (2.47), (2.48) and (2.43) respectively in the LDP equation (2.24) yields:

$$\begin{aligned} y_p(t+j|t) = {} & \tilde{\tilde{G}}_{Nj}\Delta^n u(t+j-1) + \left[\overline{\overline{G}}_{Nj}C\hat{A} + q^{-N+j}\overline{G}_N - \overline{E}_{Nj}DB\right]\Delta^n u^f(t-1) \\ & + \left[D\bar{A}\Delta^n\overline{E}_{Nj} + q^{-N+j}F_N\right]y^f(t) \end{aligned} \tag{2.50}$$

rearranging

$$\begin{aligned} y_p(t+j|t) = {} & \tilde{\tilde{G}}_{Nj}\Delta^n u(t+j-1) + \overline{\overline{G}}_{Nj}\Delta^n u(t-1) + \overline{G}_N\Delta^n u^f(t-N+j-1) \\ & + F_N y^f(t-N+j) + D\bar{A}\Delta^n\overline{E}_{Nj}y^f(t) - \overline{E}_{Nj}DB\Delta^n u^f(t-1) \end{aligned} \tag{2.51}$$

or

$$\begin{aligned} y_p(t+j|t) = {} & \tilde{\tilde{G}}_{Nj}\Delta^n u(t+j-1) + \overline{\overline{G}}_{Nj}\Delta^n u(t-1) + \overline{G}_N\Delta^n u^f(t-N+j-1) \\ & + F_N y^f(t-N+j) + \overline{E}_{Nj}\frac{DA\Delta^n y(t) - DB\Delta^n u(t-1)}{C\hat{A}} \end{aligned} \tag{2.52}$$

which using (2.40) gives:

$$\begin{aligned} y_p(t+j|t) = {} & \tilde{\tilde{G}}_{Nj}\Delta^n u(t+j-1) + \overline{\overline{G}}_{Nj}\Delta^n u(t-1) + \overline{G}_N\Delta^n u^f(t-N+j-1) \\ & + F_N y^f(t-N+j) + \overline{E}_{Nj}e(t) \end{aligned} \tag{2.53}$$

which is exactly identical to the LDOP (2.41)

◆◆◆

Remark 2.4-6:

The forced response in the LDOP is independent of the noise model. This follows because the forced response of LDOP is exactly equal to that of LDP which has already proven to be independent of noise model (see Remark 2.2-3)

2.5 The Control Law

The purpose of a long-range predictive controller is to find the current (at time = t) control action, $u(t)$, by considering the effect of current plus future control moves on the output over a future horizon bounded by times $t+N_1$ and $t+N_2$. Time $t+N_1$ is greater than or equal to the earliest time in the future at which the output is affected by the control move at time t, while N_2 is arbitrary but must be greater than N_1.

To date most of the popular long-range predictive controllers use the following type of cost function for optimal control calculation (Clarke et al., 1987):

$$J=\sum_{j=N_1}^{N_2}\gamma(j)\left[y_p(t+j|t)-w(t+j)\right]^2+\sum_{j=N_1}^{N_u}\lambda(j)\left[\Delta^{*}u(t+j-1)\right]^2 \tag{2.54}$$

where

$\gamma(.)$ = the weighting on tracking error.

$y_p(.)$ = the predicted output values.

$w(.)$= the desired output values or the set-points.

N_1 = the initial output horizon ≥ the earliest future output that is affected by the control move at time t.

N_2 = the final output horizon, the farthest future output that is included in the cost function J.

$\lambda(.)$ = the control weighting sequence.

N_u = the control horizon, the number of future non-zero control increments.

The future predicted output, $y_p(t+j|t)$, in the above cost function is usually given by the LDP or the LDOP developed above which are repeated here for reference:

$$y_p(t+j|t) = \tilde{G}_j \Delta^n u(t+j-1) + \bar{G}_j \Delta^n u^f(t-1) + F_j y^f(t) \tag{2.24}$$

LDP

$$\begin{aligned} y_p(t+j|t) = {} & \bar{\bar{\tilde{G}}}_{Nj} \Delta^n u(t+j-1) + \bar{\bar{G}}_{Nj} \Delta^n u(t-1) + \bar{G}_N \Delta^n u^f(t-N+j-1) \\ & + F_N y^f(t-N+j) + \bar{E}_{Nj} e(t) \end{aligned} \tag{2.41}$$

LDOP

The prediction consists of two parts. The first term on the RHS of the predictors is the response due to the future input moves. This first part is known as the *forced response*. The remaining terms on the RHS of the predictors are called the *free-response* i.e. the response due to the past inputs $u(.)$ and disturbances up to time t.

The forced responses are given by:

$$\text{Forced Response for LDP } = \tilde{G}_j \Delta^n u(t+j-1) \tag{2.55}$$

$$\text{Forced Response for LDOP } = \bar{\bar{\tilde{G}}}_{Nj} \Delta^n u(t+j-1) \tag{2.56}$$

where

$$\bar{\bar{\tilde{G}}}_{Nj} = \tilde{G}_j = g_0 + g_1 q^{-1} + \ldots + g_{j-1} q^{-j+1} \tag{2.57}$$

The free responses are given by:

LDP free response:

$$f(t+j) = \bar{G}_j \Delta^n u^f(t-1) + F_j y^f(t) \tag{2.58}$$

LDOP free response:

$$f(t+j) = \bar{\bar{G}}_{Nj} \Delta^n u(t-1) + \bar{G}_N \Delta^n u^f(t-N+j-1) + F_N y^f(t-N+j) + \bar{E}_{Nj} e(t) \tag{2.59}$$

In order to obtain a compact vector-matrix representation of the set of future predictions in the form:

$$\mathbf{y_p} = \mathbf{Gu} + \mathbf{f} \tag{2.60}$$

the set of N_2-N_1+1 output predictions at time t is represented by a vector $\mathbf{y_p}$ and the set of future control moves (i.e. $\Delta^n u(t)$ to $\Delta^n u(t+N_2-N_1)$) by $\mathbf{u}$. In order to have a correct vector-matrix form the polynomials $\tilde{G}_j$ are flipped right side left and the matrix produced by stacking the flipped polynomials is called **G**. Similarly the vector for the free responses is represented by **f**. In expanded form (2.60) is:

$$\begin{bmatrix} y_p(t+N_1) \\ y_p(t+N_1+1) \\ \vdots \\ \vdots \\ y_p(t+N_2) \end{bmatrix} = \begin{bmatrix} g_{N_1-1} & \cdots & g_0 & 0 & \cdots & 0 \\ g_{N_1} & \cdots & g_1 & g_0 & \ddots & 0 \\ \vdots & \vdots & \vdots & & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ g_{N_2-1} & g_{N_2-2} & \cdots & \cdots & \cdots & g_0 \end{bmatrix} \begin{bmatrix} \Delta^n u(t) \\ \Delta^n u(t+1) \\ \vdots \\ \vdots \\ \Delta^n u(t+N_2-1) \end{bmatrix} + \begin{bmatrix} f(t+N_1) \\ f(t+N_1+1) \\ \vdots \\ \vdots \\ f(t+N_2) \end{bmatrix} \tag{2.61}$$

with

$$\mathbf{y_p} = \begin{bmatrix} y_p(t+N_1) \\ y_p(t+N_1+1) \\ \vdots \\ \vdots \\ y_p(t+N_2) \end{bmatrix}_{N_2-N_1+1} \qquad \mathbf{u} = \begin{bmatrix} \Delta^n u(t) \\ \Delta^n u(t+1) \\ \vdots \\ \vdots \\ \Delta^n u(t+N_2-1) \end{bmatrix}_{N_2} \qquad \mathbf{f} = \begin{bmatrix} f(t+N_1) \\ f(t+N_1+1) \\ \vdots \\ \vdots \\ f(t+N_2) \end{bmatrix}_{N_2-N_1+1} \tag{2.62}$$

$$\mathbf{G} = \begin{bmatrix} g_{N_1-1} & \cdots & g_0 & 0 & \cdots & 0 \\ g_{N_1} & \cdots & g_1 & g_0 & \ddots & 0 \\ \vdots & \vdots & \vdots & & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ g_{N_2-1} & g_{N_2-2} & \cdots & \cdots & \cdots & g_0 \end{bmatrix}_{(N_2-N_1+1)\times N_2} \tag{2.63}$$

The dimensions of the matrix **G** are $(N_2-N_1+1)\times N_2$, where the length of **u** is N_2. However most LRPC's assume that only the first $N_u < N_2$ control moves are non-zero. This assumption greatly reduces the computational load as the dimensions of the **G** matrix reduce to $(N_2-N_1+1)\times N_u$, and the length of vector **u** becomes N_u. The **G** matrix becomes:

$$\mathbf{G} = \begin{bmatrix} g_{N_1-1} & \cdots & \cdots & g_0 & \cdots & 0 \\ g_{N_1} & \cdots & g_2 & g_1 & \ddots & 0 \\ \vdots & \vdots & \vdots & & \cdots & g_0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ g_{N_2-1} & g_{N_2-2} & \cdots & \cdots & \cdots & g_{N_2-N_u} \end{bmatrix}_{(N_2-N_1+1)\times N_u} \tag{2.64}$$

The cost function J (2.54) in vector form is:

$$J = (\mathbf{y}_p - \mathbf{w})^T \Gamma (\mathbf{y}_p - \mathbf{w}) + \mathbf{u}^T \Lambda \mathbf{u} \tag{2.65}$$

Γ= a N_2-N_1+1 by N_2-N_1+1 diagonal matrix of tracking error weightings

Λ= a N_u by N_u diagonal matrix of control weightings and

w is the vector of future set points defined as follows:

$$\mathbf{w} = \begin{bmatrix} w(t+N_1) \\ w(t+N_1+1) \\ \vdots \\ \vdots \\ w(t+N_2) \end{bmatrix}_{N_2-N_1+1} \tag{2.66}$$

Now the predictor (2.60) may be substituted in (2.65) giving the following cost function:

$$J = (\mathbf{Gu} + \mathbf{f} - \mathbf{w})^T \Gamma (\mathbf{Gu} + \mathbf{f} - \mathbf{w}) + \mathbf{u}^T \Lambda \mathbf{u} \tag{2.67}$$

Minimization of the cost function in (2.67) w.r.t **u** gives (McIntosh, 1988):

$$\mathbf{u} = (\mathbf{G}^T \Gamma \mathbf{G} + \Lambda)^{-1} \mathbf{G}^T \Gamma (\mathbf{w} - \mathbf{f}) \tag{2.68}$$

The control law (2.68), with appropriate choice of process and noise models and the various tuning parameters (e.g. N_1 , N_2 , N_u and l), yields many of the classical model-based controllers reported in the literature. Some of these include the following:

- Generalized minimum variance, GMV, (Clarke and Gawthrop, 1975 and 1979, Tuffs and Clarke, 1985).
- Extended horizon adaptive controller, EHAC, (Ydstie, 1985).
- Dynamic matrix control, DMC, (Cutler, 1979).
- Generalized predictive control, GPC, (Clarke et al., 1987).
- Identification and command,IDCOM, (Richalet et al., 1978).
- Predictive control algorithm, PCA; (Bruijn et al., 1980).

2.6 Improved Dynamic Matrix Control (IDMC)

DMC is one of the most popular LRPC's used in the chemical industry. One major drawback of DMC is its slow speed of disturbance rejection. A control horizon of one in

GPC is often very effective, whereas DMC may require N_u between 3 to 5 for similar disturbance rejection speed (Clarke, 1991).

In the following formulation a simple algorithm is developed to improve the disturbance rejection speed of classical DMC and to give two degrees of freedom for tuning, i.e. the servo and regulatory responses can be tuned independently. Note that the algorithm is extracted from the generalized noise/disturbance model structure presented earlier (2.9).

2.6.1 Classical DMC Predictor using LDP

The *measured* output model for classical DMC can be written as:

$$y(t) = Hu(t-1) + \frac{1}{\Delta}e(t) = Hu(t-1) + x(t) \tag{2.69}$$

Note that (2.69) is obtained from the generalized UMPC output model (2.9) (with an OE structure) for $A = C = D = 1$, $B = H$ and $n = 1$. The process model H and the residual $x(t)$ at time t are defined as:

$$H = \sum_{i=1}^{\infty} h_i q^{-i+1} \qquad \text{where } h_i \text{ are impulse - response coefficients} \tag{2.70}$$

$$x(t) = y(t) - Hu(t-1) = y(t) - \left[\sum_{i=1}^{\infty} h_i q^{-i+1}\right] u(t-1) \tag{2.71}$$

For stable processes the summation can be truncated at T, the process settling time, giving the following expression for $x(t)$.

$$x(t) = y(t) - \left[\sum_{i=1}^{T} h_i q^{-i+1}\right] u(t-1) \tag{2.72}$$

The measured process output, $y(t)$, can also be expressed in terms of *step*-response coefficients as:

$$y(t) = S\Delta u(t-1) + \frac{1}{\Delta}e(t) = S\Delta u(t-1) + x(t) \tag{2.73}$$

where

$$S = \frac{H}{\Delta} = \sum_{i=1}^{\infty} \frac{h_i}{\Delta} q^{-i+1} = \sum_{i=1}^{\infty} s_i q^{-i+1} \qquad \text{where } s_i \text{ are the step - response coefficients}$$

$$s_i = \sum_{k=1}^{i} h_k \qquad \text{and} \qquad h_i = s_i - s_{i-1} \quad \text{with} \quad s_0 = 0 \tag{2.74}$$

$$x(t) = y(t) - S\Delta u(t-1) = y(t) - \left[\sum_{i=1}^{\infty} s_i q^{-i+1} \right] \Delta u(t-1) \tag{2.75}$$

Again for stable processes the summation can be truncated at T provided that an additional term is included to account for the effect of past input beyond the settling time. The following expression for $x(t)$ is obtained:

$$x(t) = y(t) - \left[\sum_{i=1}^{T} s_i q^{-i+1} \right] \Delta u(t-1) - s_T u(t-T-1) \tag{2.76}$$

Note that $s_i = s_T$ for $i \geq T+1$. The Diophantine equation (2.16) becomes:

$$\frac{1}{\Delta} = E_j + q^{-j} \frac{F_j}{\Delta} = \frac{1-q^{-j}}{\Delta} + q^{-j} \frac{1}{\Delta} \tag{2.77}$$

which implies

$$E_j = \frac{1-q^{-j}}{\Delta} \qquad \text{and} \qquad F_j = 1 \tag{2.78}$$

The LHS of Diophantine equation (2.21) becomes:

$$\frac{E_j DB}{C\hat{A}} = E_j H = \frac{1-q^{-j}}{\Delta} H = \left[1-q^{-j}\right] S \tag{2.79}$$

$$S = \tilde{S}_j + q^{-j} \bar{S}_j \qquad \tilde{S}_j = \sum_{i=1}^{j} s_i q^{-i+1} \qquad \bar{S}_j = \sum_{i=j+1}^{\infty} s_i q^{-i-j+1} \tag{2.80}$$

$$\left[1-q^{-j}\right] S = S - q^{-j} S = \tilde{S}_j + q^{-j} \left[\bar{S}_j - S\right] = \sum_{i=1}^{j} s_i q^{-i+1} + q^{-j} \sum_{i=j+1}^{\infty} \left(s_i - s_{i-j}\right) q^{-i+j+1} \tag{2.81}$$

$$\tilde{G}_j = \sum_{i=1}^{j} s_i q^{-i+1} \qquad \text{and} \qquad \bar{G}_j = \sum_{i=j+1}^{\infty} \left(s_i - s_{i-j}\right) q^{-i+j+1} \tag{2.82}$$

Substituting the above in the LDP equation (2.24) gives:

$$y_p(t+j|t) = \left[\sum_{i=1}^{j} s_i q^{-i+1}\right]\Delta u(t+j-1) + \left[\sum_{i=j+1}^{\infty}\left(s_i - s_{i-j}\right)q^{-i+j+1}\right]\Delta u(t-1) + y(t) \quad (2.83)$$

$$y_p(t+j|t) = \left[\sum_{i=1}^{j} s_i \Delta u(t+j-i)\right] + \left[\sum_{i=j+1}^{\infty}\left(s_i - s_{i-j}\right)\Delta u(t-i+j)\right] + y(t) \quad (2.84)$$

Truncating the second sum at T gives the following DMC predictor:

$$y_p(t+j|t) = \left[\sum_{i=1}^{j} s_i \Delta u(t+j-i)\right] + \left[\sum_{i=j+1}^{T}\left(s_i - s_{i-j}\right)\Delta u(t-i+j)\right] + y(t) \quad (2.85)$$

Using (2.75), the DMC predictor in terms of residual $x(t)$ is obtained as:

$$y_p(t+j|t) = \left[\sum_{i=1}^{j} s_i \Delta u(t+j-i)\right] + \left[\sum_{i=j+1}^{\infty} s_i \Delta u(t+j-i)\right] + x(t) \quad (2.86)$$

which after truncation becomes:

$$\boxed{y_p(t+j|t) = \left[\sum_{i=1}^{j} s_i \Delta u(t+j-i)\right] + \left[\sum_{i=j+1}^{T} s_i \Delta u(t+j-i)\right] + s_T u(t+j-T-1) + x(t)} \quad (2.87)$$

2.6.2 Improved DMC (IDMC) Predictor using LDP

As discussed earlier in this chapter and will be shown later by simulations, an optional polynomial in the denominator of the noise/disturbance model can substantially increase the disturbance rejection speed. The original formulation of DMC lacks this option. In order to improve the disturbance rejection properties of DMC, its measured output model (2.69) is (slightly) modified to include a first order D polynomial as:

$$y(t) = Hu(t-1) + \frac{1}{D\Delta}e(t) = Hu(t-1) + \frac{1}{\left[1-\delta q^{-1}\right]\left[1-q^{-1}\right]}e(t) \quad (2.88)$$

Note that in this case $A = 1$, $C = 1$, $D = \left[1-\delta q^{-1}\right]$, $B = H$ and $n = 1$. Then (2.88) can be rewritten as:

$$y(t) = Hu(t-1) + \left[\frac{\alpha}{\Delta} + \frac{\beta}{\left[1-\delta q^{-1}\right]}\right]e(t) \quad (2.89)$$

where

$$\alpha = \frac{1}{1-\delta} \qquad \text{and} \qquad \beta = \frac{-\delta}{1-\delta} = \frac{\delta}{\delta - 1} \tag{2.90}$$

The Diophantine equation (2.16) for partitioning the noise terms becomes:

$$\left[\frac{\alpha}{\Delta} + \frac{\beta}{\left[1-\delta q^{-1}\right]}\right] = \left\{\frac{\alpha\left[1-q^{-j}\right]}{\Delta} + \frac{\beta\left[1-\delta^{j}q^{-j}\right]}{\left[1-\delta q^{-1}\right]}\right\} + q^{-j}\left\{\frac{\alpha\left[1-\delta q^{-1}\right]+\beta\delta^{j}\Delta}{\left[1-\delta q^{-1}\right]\Delta}\right\} \tag{2.91}$$

which implies

$$E_j = \left\{\frac{\alpha\left[1-q^{-j}\right]}{\Delta} + \frac{\beta\left[1-\delta^{j}q^{-j}\right]}{\left[1-\delta q^{-1}\right]}\right\} \tag{2.92}$$

and

$$F_j = \alpha\left[1-\delta q^{-1}\right] + \beta\delta^{j}\Delta = (\alpha + \beta\delta^{j}) - (\alpha\delta + \beta\delta^{j})q^{-1} \tag{2.93}$$

The LHS of Diophantine equation (2.21) becomes:

$$\begin{aligned} E_j DH &= \left\{(\alpha+\beta) - (\alpha\delta+\beta)q^{-1} - (\alpha+\beta\delta^{j})q^{-j} + (\alpha\delta+\beta\delta^{j})q^{-j-1}\right\}S \\ &= \left\{1 - (\alpha+\beta\delta^{j})q^{-j} + (\alpha\delta+\beta\delta^{j})q^{-j-1}\right\}S \end{aligned} \tag{2.94}$$

Substituting (2.74) for S gives:

$$E_j DH = \sum_{i=1}^{j} s_i q^{-i+1} + q^{-j}\sum_{i=j+1}^{\infty}\left\{s_i - (\alpha+\beta\delta^{j})s_{i-j} + (\alpha\delta+\beta\delta^{j})s_{i-j-1}\right\}q^{-i+j+1} \tag{2.95}$$

Comparing (2.95) with (2.21) implies

$$\widetilde{G}_j = \sum_{i=1}^{j} s_i q^{-i+1} \quad \text{and} \quad \overline{G}_j = \sum_{i=1}^{\infty}\left\{s_{i+j} - (\alpha+\beta\delta^{j})s_i + (\alpha\delta+\beta\delta^{j})s_{i-1}\right\}q^{-i+1} \tag{2.96}$$

Substituting the above in the LDP equation (2.24) gives:

$$\begin{aligned} y_p(t+j|t) = &\left[\sum_{i=1}^{j} s_i q^{-i+1}\right]\Delta u(t+j-1) \\ &+\left[\sum_{i=j+1}^{\infty}\left\{s_i - (\alpha+\beta\delta^{j})s_{i-j} + (\alpha\delta+\beta\delta^{j})s_{i-j-1}\right\}q^{-i+j+1}\right]\Delta u(t-1) \\ &+(\alpha+\beta\delta^{j})y(t) - (\alpha\delta+\beta\delta^{j})y(t-1) \end{aligned} \tag{2.97}$$

Applying q^{-1}, the backward shift operator, to $\Delta u(.)$'s

$$\begin{aligned} y_p(t+j|t) &= \left[\sum_{j=1}^{j} s_i \Delta u(t-i+j)\right] \\ &+\left[\sum_{i=j+1}^{\infty}\left\{s_i-(\alpha+\beta\delta^j)s_{i-j}+(\alpha\delta+\beta\delta^j)s_{i-j-1}\right\}\Delta u(t-i+j)\right] \\ &+(\alpha+\beta\delta^j)y(t)-(\alpha\delta+\beta\delta^j)y(t-1) \end{aligned} \tag{2.98}$$

defining

$$\Psi_j=(\alpha+\beta\delta^j)=\frac{1-\delta^{j+1}}{1-\delta}=\sum_{k=0}^{j}\delta^k \quad \text{and} \quad \Psi_{j-1}=(\alpha+\beta\delta^{j-1})=\frac{1-\delta^j}{1-\delta}=\sum_{k=0}^{j-1}\delta^k \tag{2.99}$$

yields

$$\begin{aligned} y_p(t+j|t) &= \left[\sum_{j=1}^{j} s_i \Delta u(t-i+j)\right]+\left[\sum_{i=j+1}^{\infty}\left\{s_i-\Psi_j s_{i-j}+\delta\Psi_{j-1}s_{i-j-1}\right\}\Delta u(t-i+j)\right] \\ &+\Psi_j y(t)-\delta\Psi_{j-1}y(t-1) \end{aligned} \tag{2.100}$$

Collecting terms in Ψ_j and Ψ_{j-1} gives

$$\begin{aligned} y_p(t+j|t) &= \left[\sum_{j=1}^{j} s_i \Delta u(t+j-i)\right]+\left[\sum_{i=j+1}^{\infty}\left(s_i-s_{i-j}\right)\Delta u(t-i+j)\right]+y(t) \\ &-\left[\sum_{i=j+1}^{\infty}\left\{(\Psi_j-1)s_{i-j}-\delta\Psi_{j-1}s_{i-j-1}\right\}\Delta u(t-i+j)\right] \\ &+(\Psi_j-1)y(t)-\delta\Psi_{j-1}y(t-1) \end{aligned} \tag{2.101}$$

rearranging gives

$$\begin{aligned} y_p(t+j|t) &= \left[\sum_{j=1}^{j} s_i \Delta u(t+j-i)\right]+\left[\sum_{i=j+1}^{\infty}\left(s_i-s_{i-j}\right)\Delta u(t-i+j)\right]+y(t) \\ &-\delta\Psi_{j-1}\left[\sum_{i=j+1}^{\infty}\left\{s_{i-j}-s_{i-j-1}\right\}\Delta u(t-i+j)\right]+\delta\Psi_{j-1}\Delta y(t) \end{aligned} \tag{2.102}$$

Truncating the second and third summations at T gives the following IDMC predictor:

$$\begin{aligned} y_p(t+j|t) &= \left[\sum_{i=1}^{j} s_i \Delta u(t+j-i)\right] + \left[\sum_{i=j+1}^{T} (s_i - s_{i-j}) \Delta u(t-i+j)\right] + y(t) \\ &\quad -\delta\Psi_{j-1}\left[\sum_{i=j+1}^{T} \{s_{i-j} - s_{i-j-1}\} \Delta u(t-i+j)\right] + \delta\Psi_{j-1}\Delta y(t) \end{aligned} \tag{2.103}$$

The above IDMC predictor is obtained in terms of residual $x(t)$ using equation (2.75) as:

$$y_p(t+j|t) = \left[\sum_{i=1}^{j} s_i \Delta u(t+j-i)\right] + \left[\sum_{i=j+1}^{\infty} s_i \Delta u(t-i+j)\right] + x(t) + \delta\Psi_{j-1}\Delta x(t) \tag{2.104}$$

which after truncation gives:

$$\boxed{\begin{aligned} y_p(t+j|t) = \left[\sum_{i=1}^{j} s_i \Delta u(t+j-i)\right] + \left[\sum_{i=j+1}^{T} s_i \Delta u(t-i+j)\right] + s_T u(t+j-T-1) + x(t) \\ +\delta\Psi_{j-1}\Delta x(t) \end{aligned}}$$

(2.105)
IDMC

Note that the "improved DMC" of (2.105) is identical to conventional DMC (2.87) except for the addition of the last term. It is shown below that this last term provides a second degree of freedom for tuning the response to disturbances. Recall that δ is the parameter in the first order D polynomial in (2.88) and Ψ_{j-1} is given by (2.99). Moreover δ varies between 0 to 1, with $\delta = 0$ giving the conventional DMC.

The residual terms in (2.105) can be viewed as Taylor series expansion of a generalized j-step ahead projected residual, $x_p(t+j|t)$, given as:

$$x_p(t+j|t) = x(t) + \xi_1 \Delta x(t) + \xi_2 \Delta^2 x(t) + \cdots \tag{2.106}$$

where ξ_i are ith partial derivative of $x_p(t+j|t)$, $\dfrac{\partial^i x_p(t+j|t)}{\partial x(t)^i}$. In the IDMC predictor (2.105) all ξ_i except ξ_0 and ξ_1 are zero (note that ξ_0 is 1). This is because of the fact that the employed D polynomial in the noise model is of first order. In general, however, there would be as many non-zero ξ_i as the order of the D polynomial. Note that the first coefficient of the expansion, ξ_0, is non-zero because of the presence of an integrator in the noise model.

2.7 Simulation Examples

In the following examples the above developed formulations/techniques are simulated using Matlab. These examples were motivated by the following propositions:

- In many cases LRPC's with a GPC noise model tends to give undesirably fast disturbance rejection leading to unacceptably excessive overshoot.
- In some LRPC applications, e.g. in the presence of unmodelled dynamics, the GPC noise model destabilizes the system.
- Classical DMC gives very slow disturbance rejection.
- More than one integrator is needed to achieve an offset-free response for ramp disturbance rejection or setpoint tracking.
- IDMC (i.e. classical DMC with a simple first order choice of the $D(q^{-1})$ polynomial) gives disturbance rejection that can be independently tuned by adjusting δ.

Simulations are based on the following two processes (Process-A and Process-B):

$$\text{Process - A (in } s\text{ - domain)} = \frac{1}{(1+s)(1+3s)(1+5s)} \tag{2.107}$$

$$\text{Process - A (in z - domain)} = \frac{.00768z^{-1} + .02123z^{-2} + .00357z^{-3}}{1 - 1.90031z^{-1} + 1.1514z^{-2} - .2158z^{-3}} \tag{2.108}$$

The poles of Process-A in the z-domain are 0.8187, 0.7165 and 0.3679, while the zeros are at -2.586 and -0.1798. Note that one of the zeros is outside the unit circle making the discretized process nonminimum phase (NMP). The gain of the process is 1.

$$\text{Process - B (in } s\text{ - domain)} = \frac{2(229)}{(s+1)(s^2+30s+229)} \tag{2.109}$$

$$\text{Process - B (in z - domain)} = \frac{.03700z^{-1} + .07172z^{-2} + .00785z^{-3}}{1 - 1.3422z^{-1} + .4455z^{-2} - .0450z^{-3}} \tag{2.110}$$

Process-B is the process used in Rohrs' (1984) benchmark example. The z-domain poles are 0.9048, 0.2187+0.0443i and 0.2187-0.0443i. Plant zeros are at -0.1164 and -1.822. Like Process-A one of the zeros is outside the unit circle making the discretized process nonminimum phase (NMP).

2.7.1 Example-1: Effect of Model Structure:

This simulation is based on Process-A with the following Reduced Order Model:

$$\text{Model - A1} = \frac{.08z^{-1}}{1 - .94z^{-1}} \tag{2.111}$$

Simulation uses N_2=10, N_1=1, N_u=1 and λ=0. A sustained step disturbance of magnitude 0.5 is introduced at time = 50.

First GPC without a T-filter (i.e. ARIX structure) is tried. The controlled variable diverged as can be seen in Figure 2.3 (note that the sustained oscillations are due to the clipping of the manipulated variable at ±5). McIntosh (1988) also reports that when using GPC without a T-filter, stable control is not possible for model-1. However changing the model structure to IOE (i.e. Integrated Output Error as in DMC), gives stable offset free control as shown in Figure 2.3. Clarke (1991) also reports that the DMC model structure, which uses a random walk noise model, is more robust than GPC without a T-filter.

One reason for the poor robustness of GPC without a T-filter is the existence of the additional factor A in the denominator of the GPC noise model. MPM is usually present at high frequencies and this A factor amplifies the modelling errors as pointed out by Clarke (1991).

Figure 2.4 demonstrates similar robustness of the DMC model structure for Process-B with the following Reduced Order Model (ROM):

$$\text{Model - B1} = \frac{.033z^{-1}}{1 - .94z^{-1}} \tag{2.112}$$

2.7.2 Example-2: Advantages of using an additional factor *D* in the Denominator of the Noise Model:

UMPC includes provision for inclusion of a polynomial factor, *D*, separate from the *A* polynomial in the noise model. Polynomial *D* can be used to increase the speed of disturbance rejection in cases when *A* is not available (i.e. for non-parameteric process models) or in the presence of high Model Plant Mismatch (MPM).

The simulations to demonstrate the role of the *D* polynomial in disturbance rejection performance use Process-A with N_2=20, N_1=1, N_u=1 and λ=0. A sustained step disturbance of magnitude 0.5 is introduced at time = 50. Figure 2.5 shows that a *D* polynomial ($D=\left[1-0.75q^{-1}\right]^2$) improves the disturbance rejection speed significantly. The improvement in disturbance rejection properties is also achieved for models with MPM as is shown in Figure 2.6 which uses $D=\left[1-0.8q^{-1}\right]\left[1-0.15q^{-1}\right]$ and the following reduced order model:

$$\text{Model - A2} = \frac{.012z^{-1}+.023z^{-2}+.065z^{-3}}{1\ -\ .91z^{-1}} \tag{2.113}$$

2.7.3 Example-3: Double Integrator for Ramp disturbances:

An integrator in the noise model is usually required for offset removal. However in some cases a single integrator may not be sufficient. As defined earlier the general integrator included in the UMPC model (2.9) may take the form of a single integrator $1/\Delta$, a double integrator $1/\Delta^2$ or a multiple integrator $1/\Delta^n$.

To illustrate the need of a double integrator, a ramp disturbance of 0.1*t* is introduced at *t*=50 . Figure 2.7 shows that for a ramp disturbance DMC noise model gives a controlled variable response with an offset. However an additional integrator gives an offset-free response. The need for a double integrator for GPC is demonstrated in Figure 2.8 where a single integrator failed to remove the steady state offset. Note that the magnitude of the offset in case of GPC is much lower than that of DMC. This is due to the presence of the additional A polynomial in GPC. Both figures are based on Process-A with no MPM.

Note that the ramp disturbances as shown in Figures 2.7 and 2.8 are much more common in process industries than might be first thought. UMPC as in most LRPC's, assumes that the disturbance adds directly to the output. Most conventional process models assume that the disturbance, d, passes through a disturbance transfer function, G_d, before it reaches the output. Process transfer functions containing an integrator are quite common in industry, e.g. most liquid level processes. Obviously a step disturbance, d, passing through an integrating G_d will appear at the output as a ramp which is the case illustrated by Figures 2.7 and 2.8. In practice, DMC applications to integrating processes are handled by differencing the process measurement which essentially adds an integrator to the controlled system.

2.7.4 Example-4: Double Integrator for Ramp Set point Tracking:

A single integrator can not give offset-free response for ramp tracking. A double integrator is required. This proposition is illustrated in Figure 2.9 which clearly shows the offset when a single integrator is used. The offset is removed when a double integrator is used. These simulations used Process-B with no unmodelled dynamics. Note that the setpoint tracking is not a function of noise model structure as long as there is no MPM.

2.7.5 Example-5: Disturbance Rejection Properties of IDMC:

Figure 2.10 compares the disturbance rejection performance of IDMC with classical DMC. The simulation is based on Process-A with no MPM. For a control horizon of 1, DMC gives poor disturbance rejection performance. The addition of $D=\left[1-0.9q^{-1}\right]$ yields a significant increase in disturbance rejection speed. Note that IDMC does not require on-line solution of a Diophantine equation to incorporate a first order D in the control law. There is simply an extra term in the predictor (2.105).

2.8 Conclusions

The Unified Model Predictive Controller (UMPC) developed in this study integrates several model structures into a single algorithm. The designer is free to change the model structure a priori or on-line, to meet the current requirements of a given application. With the conventional LDP the computational load for UMPC is slightly greater than predictive controllers based on a less-general model structure such as GPC. However the UMPC with the overparameterized predictor formulation (LDOP) substantially reduces the computational load of the controller algorithm. The generalized noise/disturbance model of UMPC yields the classical DMC and GPC formulations as a special cases. A simple modification to the classical DMC algorithm gives considerably faster disturbance rejection for a control horizon of one. The specific results and conclusion of current work are as follows:

- Model structure has a significant influence on the controller robustness and performance.
- The combination of the C and D polynomials in UMPC is more flexible than the simple T-filter (the *designed* C polynomial) used in GPC.
- A double integrator is needed for offset-free response for ramp disturbances and/or set points.
- For ARIMAX structures, UMPC is exactly the same as GPC with a T-filter.
- UMPC gives the classical DMC predictor as a special case when the generalized noise model is set equal to a simple integrator e.g. $1/\Delta$.
- An improved form of DMC for regulatory control (IDMC) is obtained by a simple derivation using the LDP and a first order lag filter in the original DMC noise model. The IDMC gives independent tuning, e. g. for a faster disturbance rejection, of the classical DMC.

References:

Box, G.E.P. and Jenkins, G. M., "Time Series Analysis, Forecasting and Control", Holden-Day, San Francisco, 1976.

Bruijn, P.M., Bootsma, L.J. and Verbruggen, H.B., "Predictive Control using Impulse response Models", IFAC Symposium on Digital Computer Applications to Process Control, Dusseldorf, 1980.

Clarke, D.W.," Adaptive Generalized Predictive Control ", Proceedings of the Fourth International Conference on Process Control (CPCIV), Editors Y. Arkun and W. H. Ray, Padre Island, TX, 1991.

Clarke, D.W. and Gawthrop. P.J., "Self-Tuning Controller", Proc. IEE, Vol. 122, No. 9, pp.929-934, 1975.

Clarke, D.W. and Gawthrop. P.J., "Self-Tuning Control", Proc. IEE, Vol. 126, No. 6, pp.633-640, 1979.

Clarke, D.W., Mohtadi, C. and Tuffs, P.S., " Generalized Predictive Control - Part I the Basic Algorithm ", Automatica, Vol. 23, No. 2, pp.137-148, 1987a.

Clarke, D.W., Mohtadi, C. and Tuffs, P.S., " Generalized Predictive Control - Part II Extensions and Interpretations ", Automatica, Vol. 23, No. 2, pp. 149-160, 1987b.

Cutler, C.R., "Dynamic Matrix Control - A Computer Control Algorithm ", AIChE National Meeting, Houston, TX, 1979.

De Keyser, M.C., "Basic Principles of Model Based Predictive Control", Proceedings of the First European Control Conference, pp. 1753-1758, 1991.

Harris T.J. and McGregor, J.F., "Design of Discrete Multivariable Linear-Quadratic Controllers using Transfer Functions", Report #1002, Department of Chemical Engineering, McMaster University, 1986.

Li, S., Lim, K.L. and Fisher, D.G., "A State Space Formulation for Model Predictive Control ", A.I.Ch.E. Journal, Vol. 35, No. 2, pp. 241-249, 1989.

Lim, K.L. "Multivariable Optimal Constrained Control Algorithm (MOCCA)", M.Sc. thesis, University of Alberta, 1988.

Ljung, L., " System Identification: Theory for the User ", Prentice Hall, Englewood Cliffs, New Jersey, 1987.

Lundstrom, P., Lee, J.H. and Morari, M., "Limitations of Dynamic Matrix Control", Proceedings of the First European Control Conference, pp. 1839-1844, 1991.

McIntosh, A.R., "Performance and Tuning of Adaptive Generalized Predictive Control ", M.Sc. thesis, University of Alberta, 1988.

Richalet, J., Rault, A., Testud, J.L., and Papon, J., "Model Predictive Heuristic Control: Applications to Industrial Processes", Automatica, Vol. 14, pp. 413, 1978.

Rohrs, C.E., Athans, M., Valavani, L. and Stein, G., "Some Design Guidelines for Discrete-time Adaptive Controllers", Automatica, Vol. 20, No. 5, pp. 653-660, 1984.

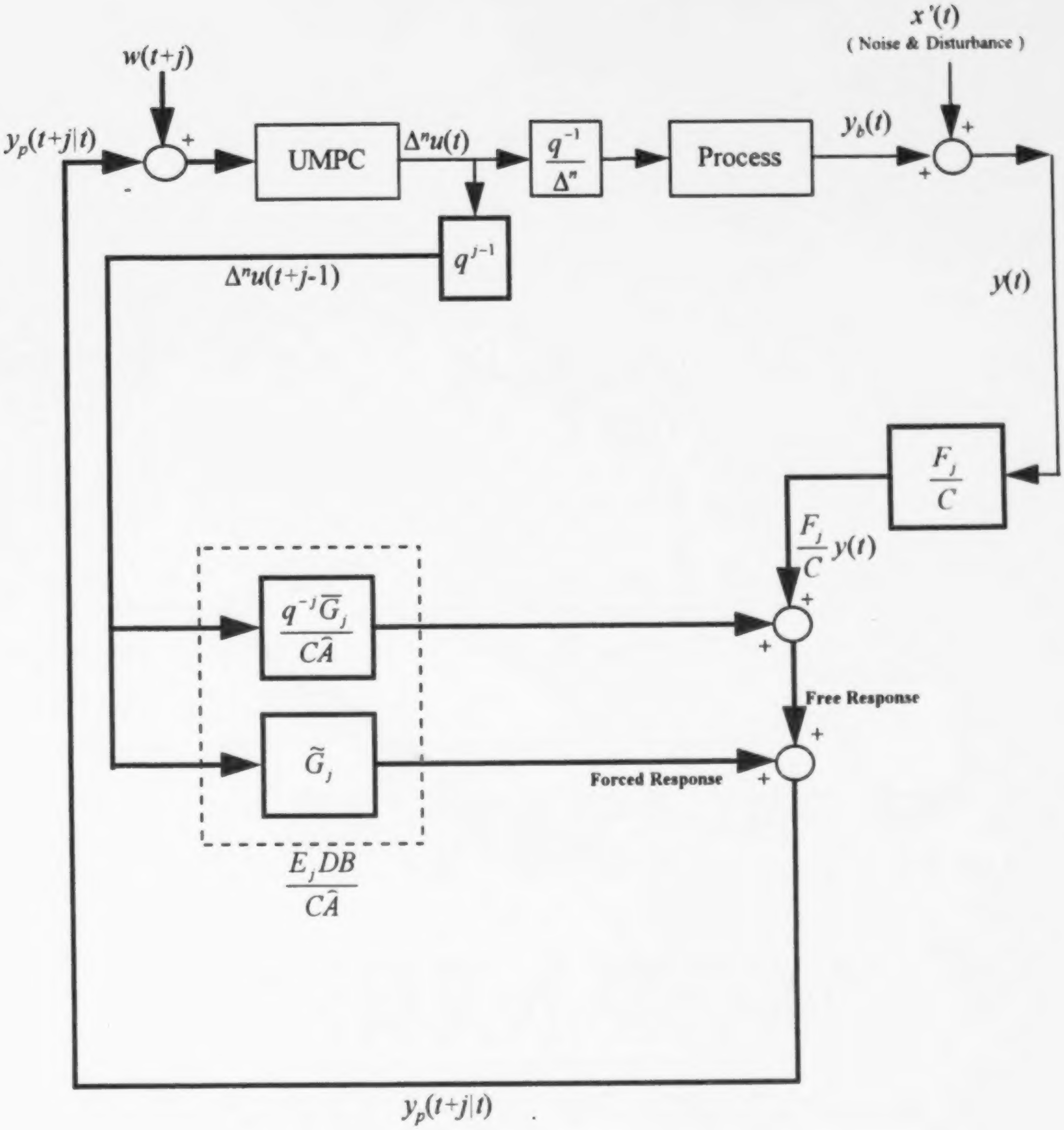
Soderstrom, T. and Stoica, P., " System Identification", Prentice Hall International Series in Systems and Control Engineering, New York, 1989.

Soeterboek, A.R.M., "Predictive Control — A Unified Approach", Ph.D. thesis, Delft University of Technology, Netherlands, 1990.

Soeterboek, A.R.M., "Predictive Control — A Unified Approach", Prentice Hall International Series in Systems and Control Engineering, 1992.

Soeterboek, A.R.M., Verbruggen, H.B., Bosch, P.P.J., and Butler,H., "On the Unification of Predictive Control Algorithms", Proceedings of the 29th Conference on Decision and Control, Honolulu, Hawaii, 1990.

Ydstie, B.E., Kershenbaum, L.S. and Sargent, R.W.H., "Theory and Applications of an Extended Horizon Self-Tuning Controller", A.I.Ch.E. Journal, Vol. 31, No. 11, pp. 1771-1780, 1985.

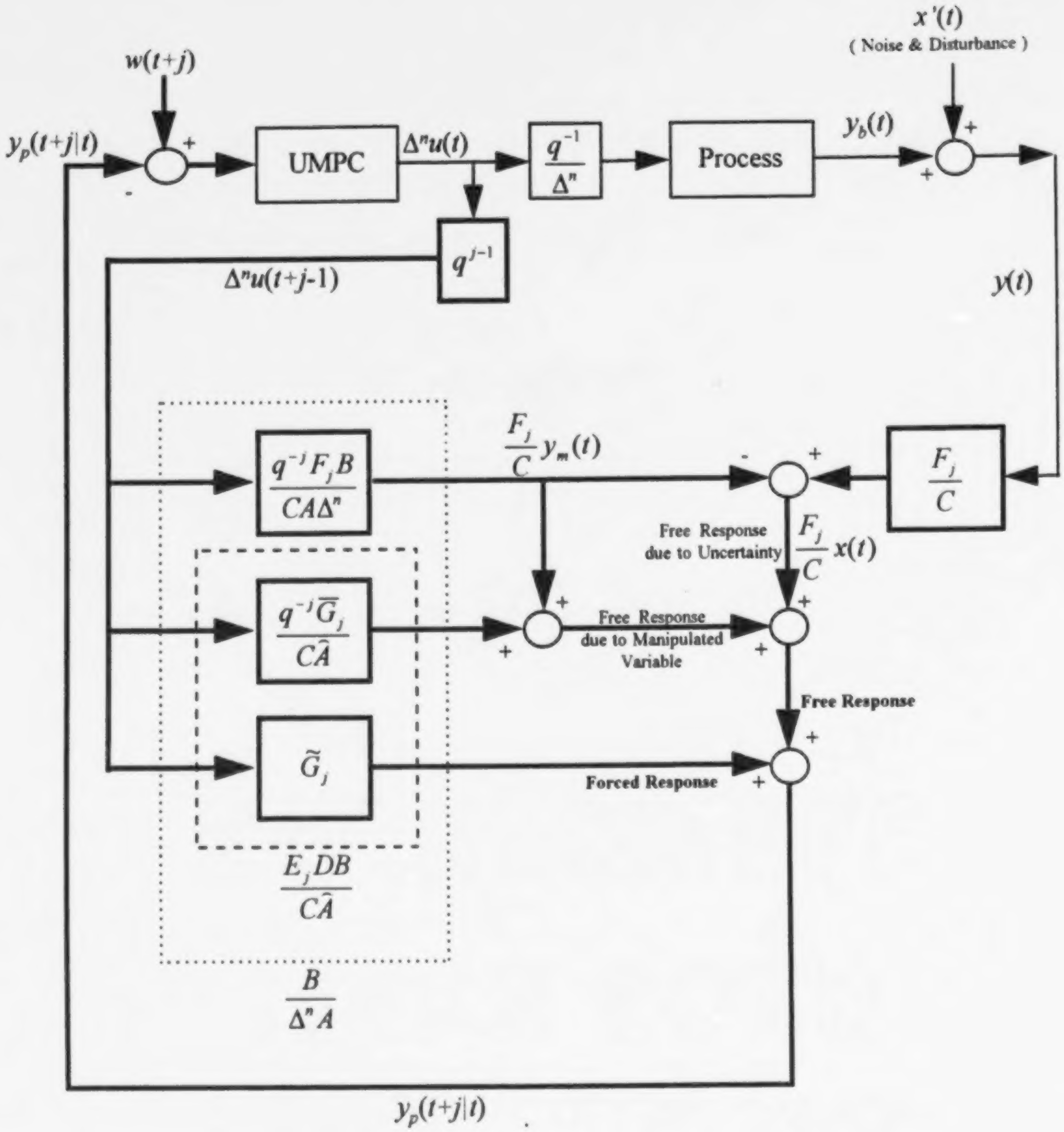

$w(t+j)$
$x'(t)$
(Noise & Disturbance)
$y_p(t+j|t)$
+
-
UMPC
$\Delta^n u(t)$
$\frac{q^{-1}}{\Delta^n}$
Process
$y_b(t)$
+
+
q^{j-1}
$\Delta^n u(t+j-1)$
$y(t)$
$\frac{F_j}{C}$
$\frac{F_j}{C} y(t)$
+
$\frac{q^{-j}\overline{G}_j}{C\hat{A}}$
+
Free Response
+
$\tilde{G}_j$
Forced Response
+
$\frac{E_j DB}{C\hat{A}}$
$y_p(t+j|t)$

Notes:

1) The filters and signals with the index j are vectors and are drawn with thicker lines; j varies from 1 to N_2.

2) The UMPC block does the calculations (optimization) necessary to produce the control vector that minimizes the predicted control errors.

Figure 2.1 Unified model predictive control (UMPC) with lumped (GPC type) Diophantine predictor (LDP)


$w(t+j)$
$x'(t)$
(Noise & Disturbance)
$y_p(t+j|t)$
+
-
UMPC
$\Delta^n u(t)$
$\frac{q^{-1}}{\Delta^n}$
Process
$y_b(t)$
+
+
q^{j-1}
$\Delta^n u(t+j-1)$
$y(t)$
$\frac{F_j}{C} y_m(t)$
$\frac{q^{-j} F_j B}{C A \Delta^n}$
-
+
$\frac{F_j}{C}$
Free Response due to Uncertainty
$\frac{F_j}{C} x(t)$
$\frac{q^{-j} \overline{G}_j}{C \hat{A}}$
+
+
Free Response due to Manipulated Variable
+
+
Free Response
$\widetilde{G}_j$
+
Forced Response
+
$\frac{E_j DB}{C \hat{A}}$
$\frac{B}{\Delta^n A}$
$y_p(t+j|t)$

Notes:

1) The filters and signals with the index j are vectors and are drawn with thicker lines; j varies from 1 to N_2.

2) The UMPC block does the calculations (optimization) necessary to produce the control vector that minimizes the predicted control errors.

Figure 2.2 Analysis of the LDP to separate free responses corresponding to the manipulated variable and the uncertainty

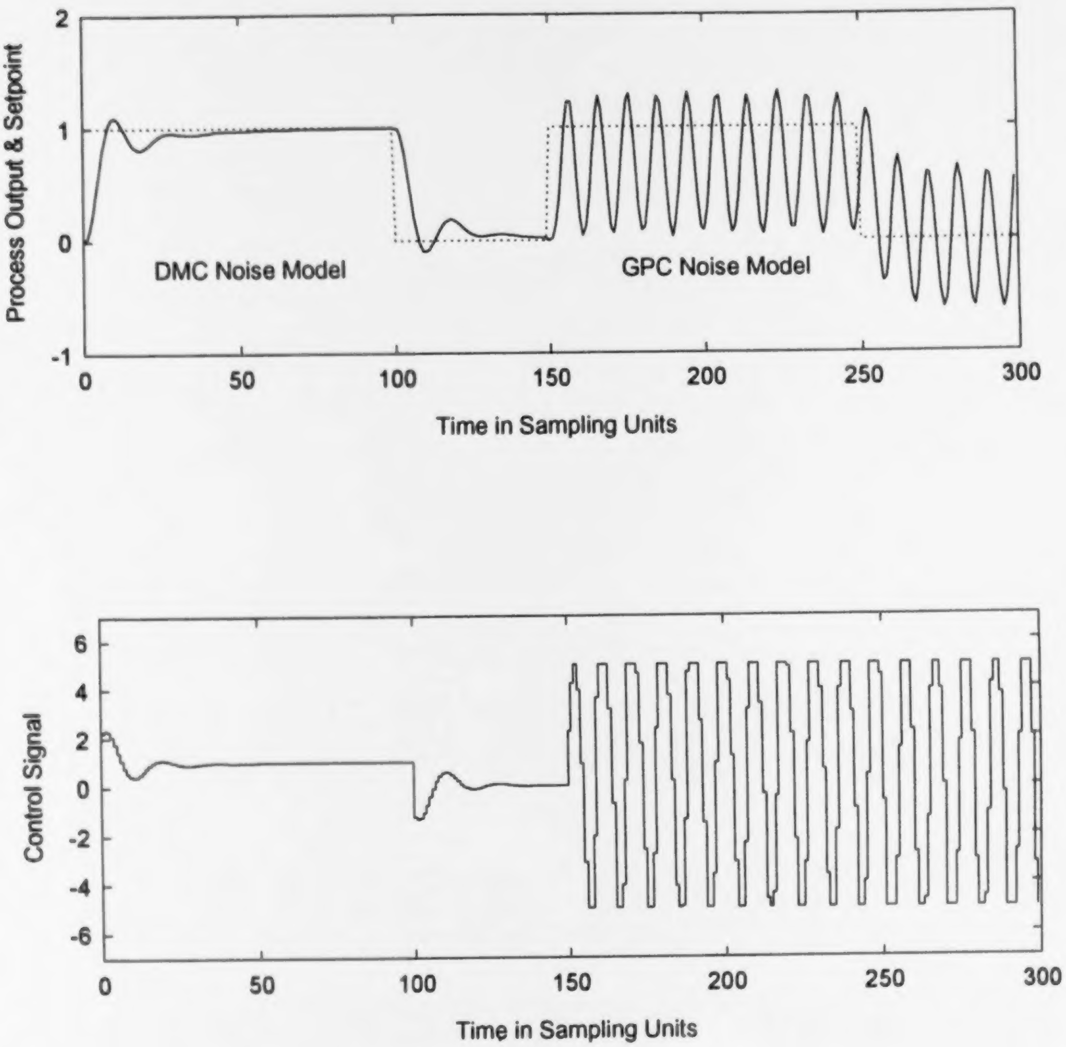


Figure 2.3 Effect of noise model in the presence of model plant mismatch (MPM)

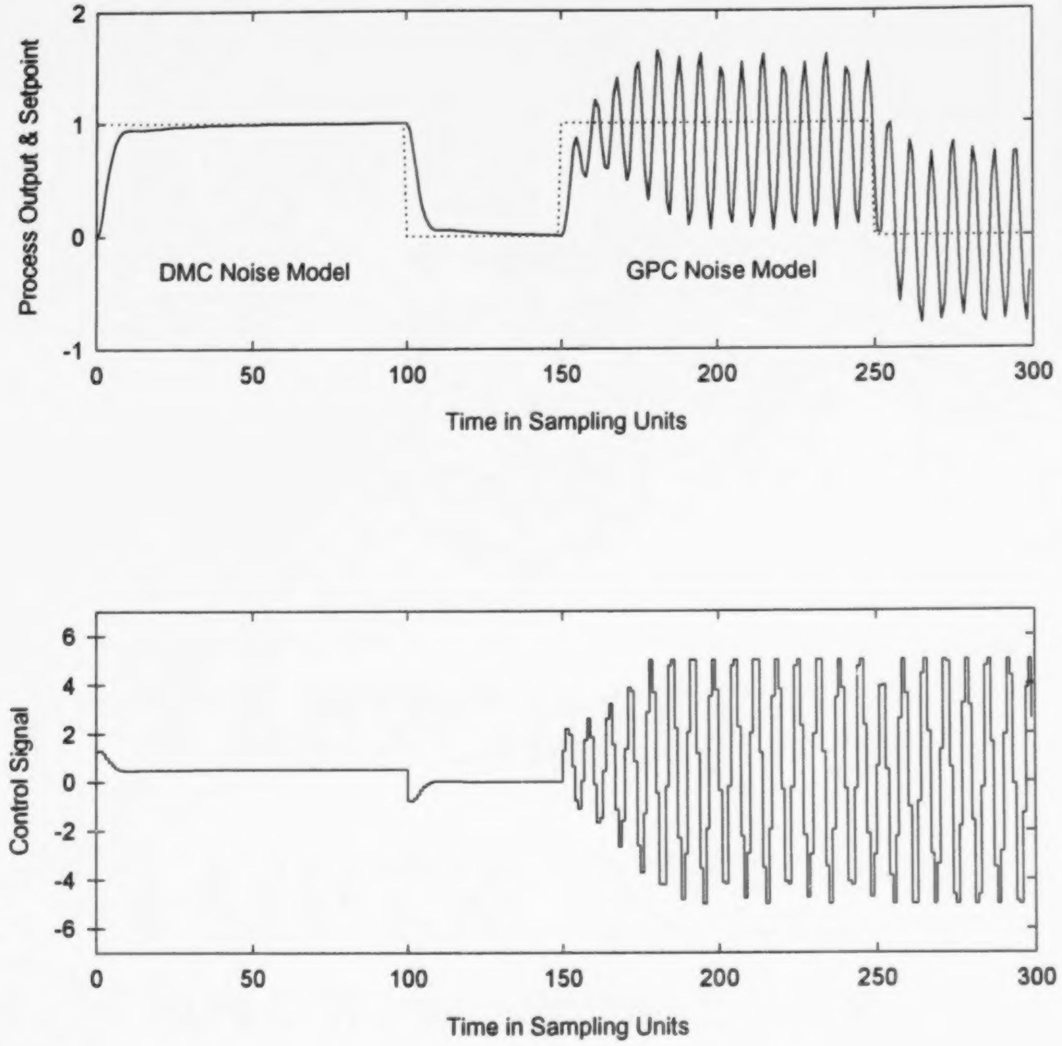


Figure 2.4 Effect of noise model in the presence of MPM for Rohr's process

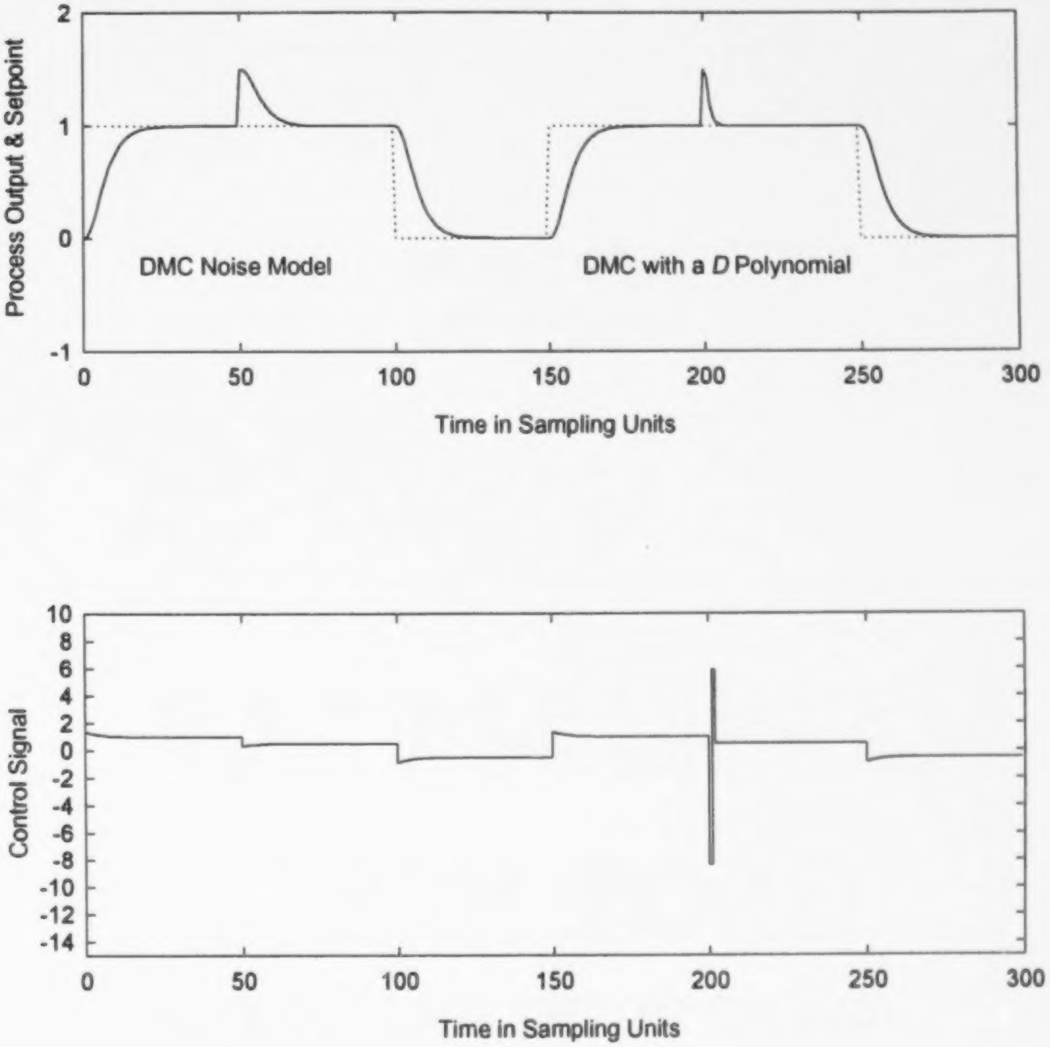


Figure 2.5 Role of *D* polynomial in disturbance rejection properties of UMPC

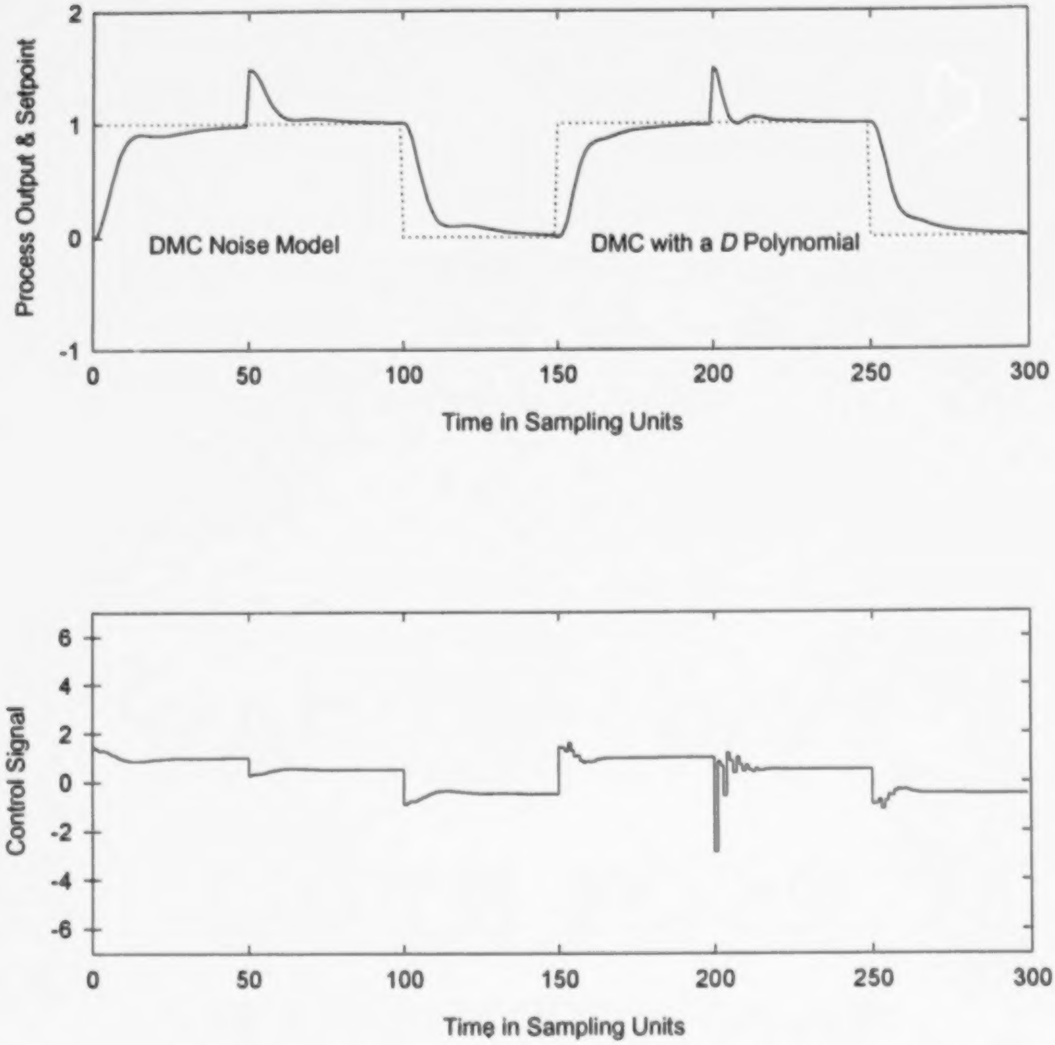


Figure 2.6 Effect of *D* polynomial in the presence of MPM

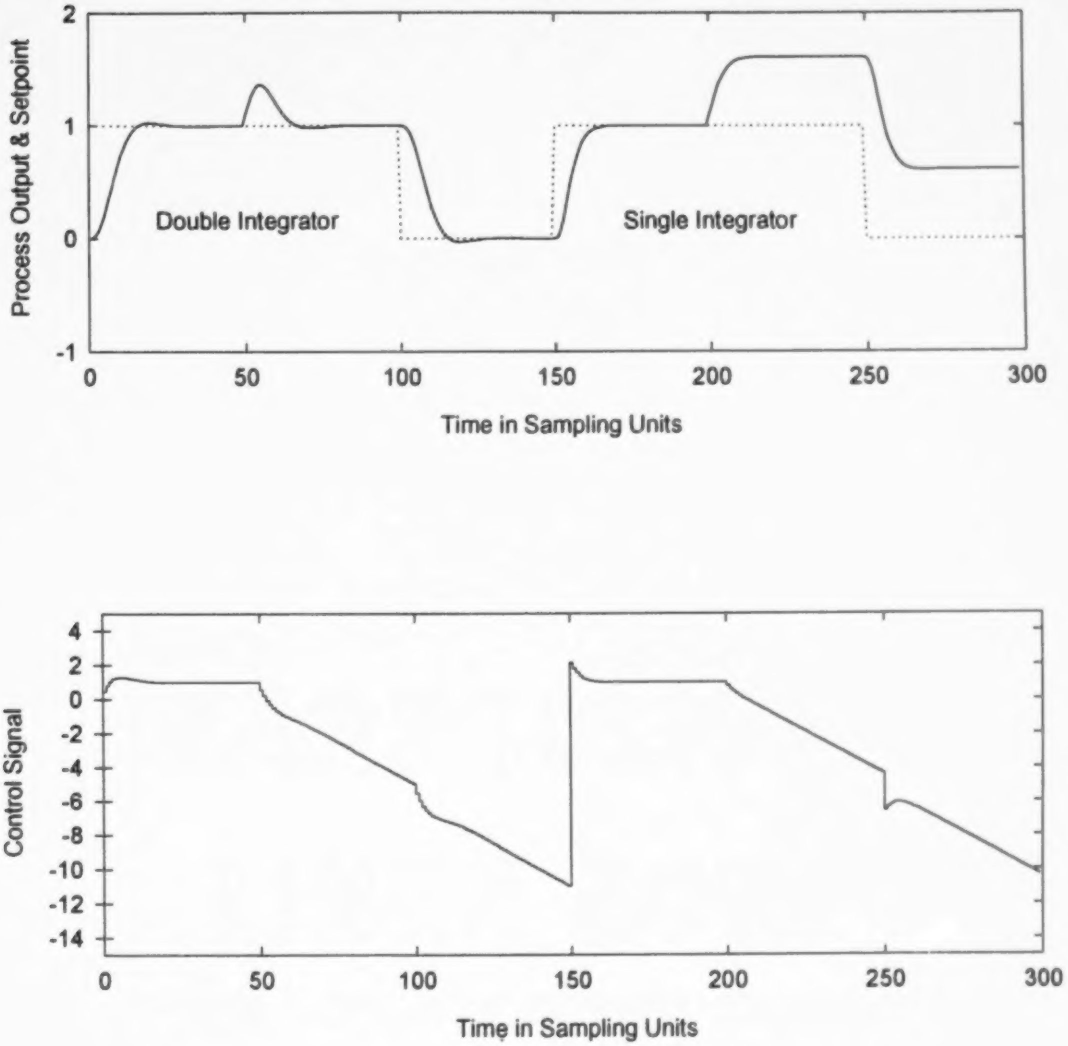


Figure 2.7 Offset removal using double integrator for ramp disturbances in DMC

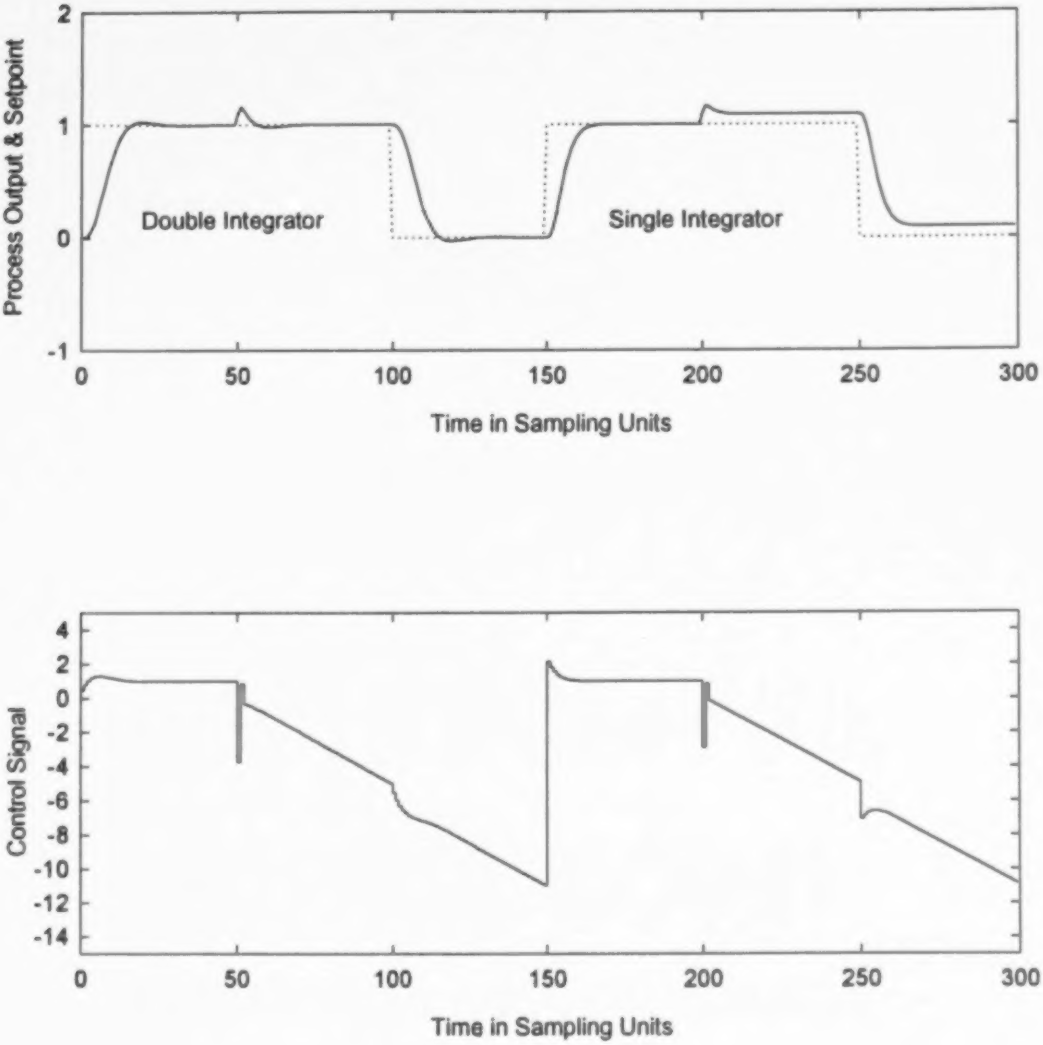


Figure 2.8 Offset removal using double Integrator for ramp disturbances in GPC

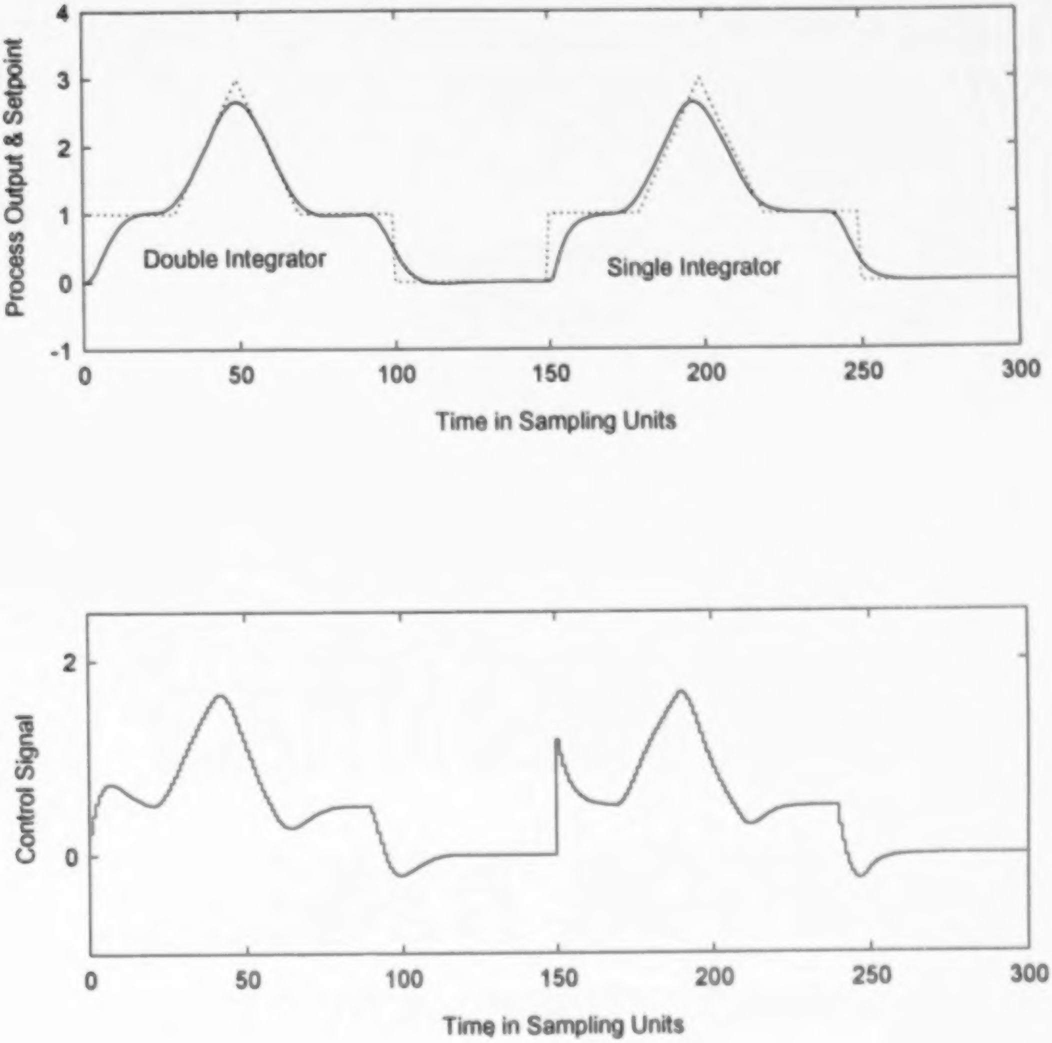


Figure 2.9 Offset-free ramp tracking using a double integrator

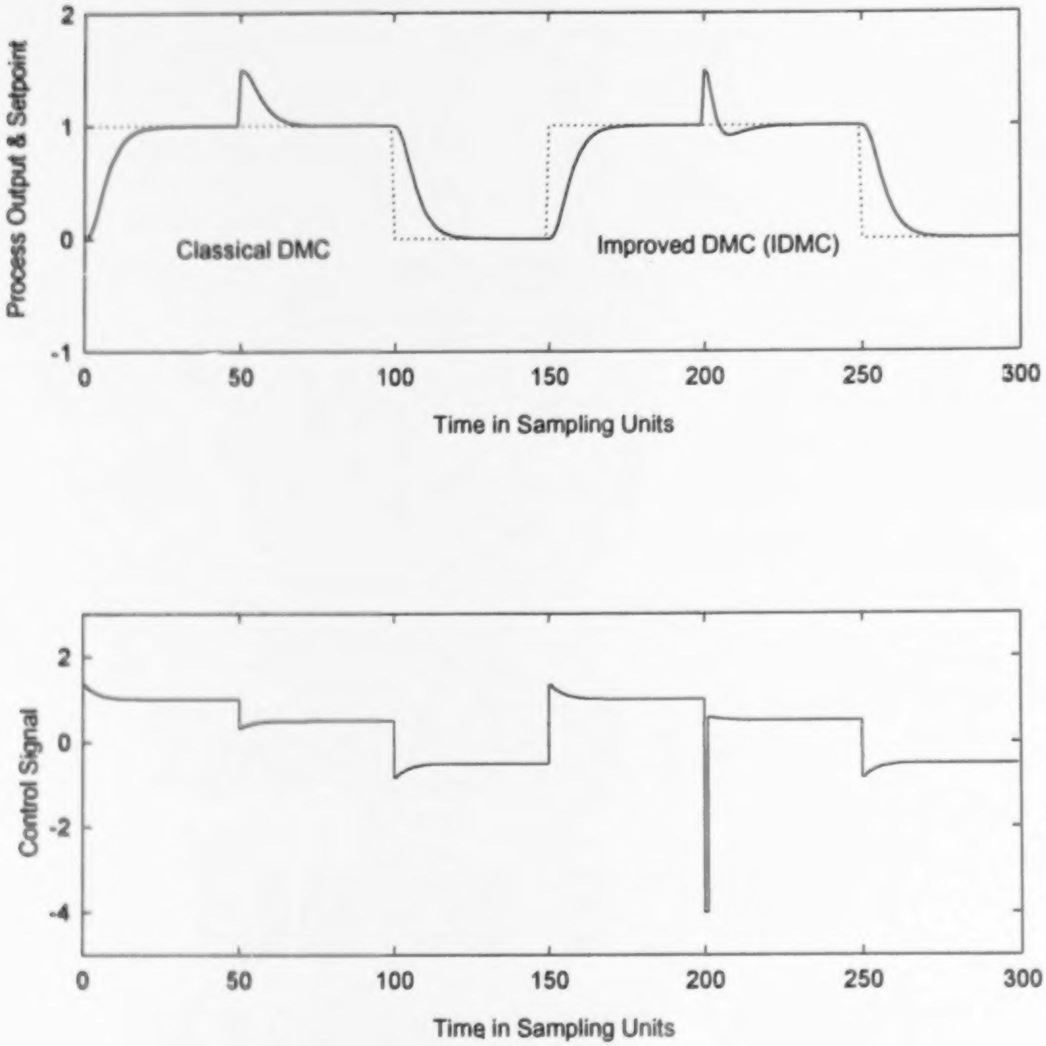


Figure 2.10 Disturbance rejection properties of improved DMC (IDMC)

Chapter 3

Disturbance Rejection Techniques for Long Range Predictive Control

3.1 Introduction

This chapter develops a new Separated Diophantine Predictor (SDP) which separates the noise model from the plant model in the prediction equation and hence makes it possible to manipulate the disturbance term independent of the plant model. The SDP is proven to be mathematically identical to the classical Lumped Diophantine Predictor (LDP) used in GPC.

An overparameterized version of the SDP, the SDOP (Separated Diophantine Overparameterized Predictor), is also developed. It is functionally identical to the SDP but offers substantial savings in computational load.

A Long Range Predictive Control (LRPC) algorithm is derived using each of the three predictors , i.e. LDP, SDP and SDOP. An improved version of Dynamic Matrix Control (DMC) is also formulated using the same concepts. A disturbance horizon to control the extent of the employed noise model is introduced and its role in improving robustness is established. Another new method, σ-weighting, which uses two (or more) noise/disturbance models with normalized weightings is also developed to provide additional flexibility for regulatory control.

3.2 Model and Predictor Preliminaries

The following derivation follows that given in Chapter 2.

The following linear discrete-time generalized model structure is used to represent the measured output:

$$y(t)=\frac{B}{A}u(t-1)+\frac{C}{D\breve{A}\Delta^{n}}e(t) \tag{3.1}$$

where the notation $\breve{A}$ is used to indicate that the polynomial A is only included for Equation Error (EE) model structures and is be omitted for Output Error (OE) structures. More specifically:

$$\breve{A}=A \qquad \text{for EE structures} \tag{3.2}$$

$$\breve{A}=1 \qquad \text{for OE structures} \tag{3.3}$$

Alternatively, the output may be expressed as

$$y(t)=\frac{B}{\breve{A}}u(t-1)+x(t) \tag{3.4}$$

where $x(t)$ is the residue at time t.

For j-step ahead predictions the time parameter in equation (3.1) is shifted by $+j$ giving:

$$y(t+j)=\frac{B}{A}u(t+j-1)+\left[\frac{C}{D\breve{A}\Delta^{n}}\right]e(t+j) \tag{3.5}$$

The noise term is separated into future and past/present components using the Diophantine equation:

$$\left[\frac{C}{D\breve{A}\Delta^{n}}\right]=E_j+q^{-j}\frac{F_j}{D\breve{A}\Delta^{n}} \tag{3.6}$$

This Diophantine expansion can also be employed in the expansion of the process model term. To do this equation (3.5) is rearranged as follows:

$$y(t+j)=\left[\frac{C}{D\breve{A}\Delta^{n}}\right]DB\frac{\Delta^{n}u(t+j-1)}{C\hat{A}}+\left[\frac{C}{D\breve{A}\Delta^{n}}\right]e(t+j) \tag{3.7}$$

Where $\hat{A}$ is defined as follows:

$$\hat{A} = 1 \qquad \text{for EE structures} \tag{3.8}$$

$$\hat{A} = A \qquad \text{for OE structures} \tag{3.9}$$

and, in general, the following relation holds:

$$\hat{A}\bar{A} = A \tag{3.10}$$

Using (3.6) to substitute for $\left[\frac{C}{D\bar{A}\Delta^n}\right]$ in (3.7) gives:

$$y(t+j) = \left[E_j + q^{-j}\frac{F_j}{D\bar{A}\Delta^n}\right]DB\frac{\Delta^n u(t+j-1)}{C\hat{A}} + \left[E_j + q^{-j}\frac{F_j}{D\bar{A}\Delta^n}\right]e(t+j) \tag{3.11}$$

Setting the future component $E_j e(t+j)$ equal to zero and rearranging gives:

$$y_p(t+j|t) = \frac{E_j DB}{C\hat{A}}\Delta^n u(t+j-1) + \frac{F_j}{C}\left[\frac{B}{A}u(t-1) + \frac{C}{D\bar{A}\Delta^n}e(t)\right] \tag{3.12}$$

Then recognizing that $\left[\frac{B}{A}u(t-1) + \frac{C}{D\bar{A}\Delta^n}e(t)\right] = y(t)$ the above equation can be simplified as follows:

$$y_p(t+j|t) = \left[\frac{E_j DB}{C\hat{A}}\right]\Delta^n u(t+j-1) + \frac{F_j}{C}y(t) \tag{3.13}$$

The first term on the RHS of (3.13) contains both, the future and the past, components. In order to separate these components the following Diophantine equation is used:

$$\left[\frac{E_j DB}{C\hat{A}}\right] = \tilde{G}_j + q^{-j}\frac{\bar{G}_j}{C\hat{A}} \tag{3.14}$$

Then (3.13) becomes

$$y_p(t+j|t) = \tilde{G}_j\Delta^n u(t+j-1) + \frac{\bar{G}_j}{C\hat{A}}\Delta^n u(t+j-1) + \frac{F_j}{C}y(t) \tag{3.15}$$

With the following definitions of filtered input and output

$$\Delta^n u^f(t-1) = \frac{\Delta^n u(t-1)}{C\hat{A}} \qquad \text{and} \qquad y^f(t) = \frac{y(t)}{C} \tag{3.16}$$

the predictor (3.15) becomes:

$$\boxed{y_p(t+j|t) = \widetilde{G}_j \Delta^n u(t+j-1) + \overline{G}_j \Delta^n u^f(t-1) + F_j y^f(t) \qquad \begin{matrix}(3.17)\\ \text{LDP}\end{matrix}}$$

In the following discussion (3.17) is referred to as the Lumped Diophantine Predictor or LDP.

3.3 Separated Diophantine Predictor (SDP)

As mentioned earlier, the noise/disturbance model provides a direct means of manipulating the disturbance rejection properties of the controller. It is therefore desirable to formulate a predictor in which the noise term can be manipulated independent of the process term. This section proposes a simple formulation to achieve this goal.

Equation (3.5) can be rearranged as:

$$y(t+j) = \left[\frac{B}{\Delta^n A}\right] \Delta^n u(t+j-1) + \left[\frac{C}{D\breve{A}\Delta^n}\right] e(t+j) \tag{3.18}$$

The first term of the above equation is separated into past and future/present components using the following Diophantine expansion:

$$\left[\frac{B}{\Delta^n A}\right] = \widetilde{P}_j + q^{-j} \frac{\overline{P}_j}{\Delta^n A} \tag{3.19}$$

The noise model $\left[\frac{C}{D\breve{A}\Delta^n}\right]$ is separated using the Diophantine expansion given in (3.6). The predictor based on these separated Diophantine equations is developed below and is referred to as Separated Diophantine Predictor or SDP.

Substituting (3.19) in (3.18) and using the Diophantine expansion given in (3.6).with future $e(.)$ set to zero gives:

$$y_p(t+j|t) = \widetilde{P}_j \Delta^n u(t+j-1) + \frac{\overline{P}_j}{A} u(t+j-1) + \frac{F_j}{D\breve{A}\Delta^n} e(t) \tag{3.20}$$

With the following definition of the filtered input and filtered error

$$u^F(t-1) = \frac{u(t-1)}{A} \qquad \text{and} \qquad e^f(t) = \frac{e(t)}{D\breve{A}\Delta^n} \tag{3.21}$$

the predictor in (3.20) becomes:

$$y_p(t+j|t) = \widetilde{P}_j \Delta^n u(t+j-1) + \overline{P}_j u^F(t-1) + F_j e^f(t) \tag{3.22}$$

The filtered error may be obtained using (3.1) as:

$$e^f(t) = \frac{e(t)}{D\breve{A}\Delta^n} = \frac{1}{C}\left[y(t) - \frac{B}{A}u(t-1)\right] = \frac{x(t)}{C} = x^f(t) \tag{3.23}$$

where

$$x(t) = \text{residual at } t = \left[y(t) - \frac{B}{A}u(t-1)\right] \tag{3.24}$$

and $x^f(t)$ is filtered residue at time t.

Note that the residual $x(t)$ is independent of the noise model and therefore the filtered error $e^f(t)$ and the filtered residue $x^f(t)$ depend only on the numerator, C, of the noise model and are independent of its denominator. The predictor can be written in terms of the filtered residual as follows:

$$\boxed{y_p(t+j|t) = \widetilde{P}_j \Delta^n u(t+j-1) + \overline{P}_j u^F(t-1) + F_j x^f(t) \qquad \begin{array}{r}(3.25)\\ \text{SDP}\end{array}}$$

A block diagram of SDP is given in Figure 3.1. The manipulated variable and the uncertainty contributions to output prediction are clearly shown. Comparing this figure with Figure 2.2 proves that the SDP formulation is much simpler than the LDP.

Remark 3.3-1:

The SDP has two parts. The first part consists of the first two terms which depend exclusively on the plant model. The second part, which is the last term in the SDP, depends on the employed noise model and the residual $x(t)$. Thus the process and noise/disturbance contributions to the output prediction are separated.

Remark 3.3-2:

When the process polynomials A and B are estimated on-line (i.e. in adaptive control) it is not necessary to solve the noise model Diophantine equation (3.6) at each control interval for OE structures, since it is independent of A and B. This results in significant computational savings.

Remark 3.3-3:

The SDP for the OE noise structure provides a basis for robustness analysis that is independent of the process model. This is possible because the noise model contribution to predictions (3.25) is free of the process model denominator and is also separately available in the predictor.

Remark 3.3-4:

The SDP is straightforward and is simpler than the LDP which lumps together the plant model and noise model terms.

Remark 3.3-5:

The SDP is mathematically equivalent to the LDP as is shown in Theorem-1.

Theorem-1:

The SDP (equation (3.25)) is mathematically identical to the LDP (equation (3.17)).

Proof:

The first step in the proof is to prove that $\tilde{G}_j = \tilde{P}_j$.

$\tilde{G}_j$ can be written as:

$$\tilde{G}_j = \left[\frac{E_j D B}{C\hat{A}}\right]_j = \left[\frac{E_j D\breve{A} B}{CA}\right]_j \tag{3.26}$$

where the subscript j following the outer square bracket indicates the first j terms of the infinite order polynomial within the outer square brackets.

Similarly $E_j(q^{-1})$ can be written as:

$$E_j = \left[\frac{C}{D\breve{A}\Delta^n}\right]_j \tag{3.27}$$

Substituting (3.27) for E_j in (3.26) gives

$$\widetilde{G}_j = \left[\frac{B}{A\Delta^n}\right]_j = \widetilde{P}_j \tag{3.28}$$

This result can be used to prove that SDP = LDP.

Substituting (3.23) for $x^f(t)$ in (3.25) gives:

$$y_p(t+j|t) = \widetilde{P}_j \Delta^n u(t+j-1) + \overline{P}_j u^F(t-1) - F_j \frac{B}{C} u^F(t-1) + F_j y^f(t) \tag{3.29}$$

Transforming $u^F(t-1)$ into $\Delta^n u^f(t-1)$ using (3.21) and (3.16) yields

$$y_p(t+j|t) = \widetilde{P}_j \Delta^n u(t+j-1) + \frac{\overline{P}_j C - F_j B}{\Delta^n \breve{A}} \Delta^n u^f(t-1) + F_j y^f(t) \tag{3.30}$$

and multiplying (3.19) throughout by C gives:

$$\frac{CB}{\Delta^n A} = C\widetilde{P}_j + q^{-j} \frac{C\overline{P}_j}{\Delta^n A} \tag{3.31}$$

$$CB = C\Delta^n A\widetilde{P}_j + q^{-j} C\overline{P}_j \tag{3.32}$$

Multiplying (3.6) throughout by B gives:

$$\frac{BC}{D\breve{A}\Delta^n} = BE_j + q^{-j} \frac{BF_j}{D\breve{A}\Delta^n} \tag{3.33}$$

or

$$BC = BD\breve{A}\Delta^n E_j + q^{-j} BF_j \tag{3.34}$$

Equating (3.32) and (3.34) gives

$$\frac{C\overline{P}_j - BF_j}{\Delta^n \breve{A}} = q^j C\hat{A}\left[\frac{BDE_j}{C\hat{A}} - \widetilde{P}_j\right] \tag{3.35}$$

and using equation (3.14) and the fact that $\widetilde{P}_j = \widetilde{G}_j$ gives:

$$\frac{C\overline{P}_j - BF_j}{\Delta^n \breve{A}} = \overline{G}_j \tag{3.36}$$

Substituting (3.36) and $\widetilde{P}_j = \widetilde{G}_j$ in (3.30) proves that the SDP is identical to the LDP.

◆ ◆ ◆

Remark 3.3-6:

For unstable systems the $u^F(.)$ and $x^f(t)$ in the SDP get unbounded even for EE noise structures, whereas the LDP in such cases gives stable predictions. An easy solution to this problem is to use the following identities to convert the SDP form into the LDP equivalent form:

$$\tilde{G}_j = \tilde{P}_j \tag{3.37}$$

$$\bar{G}_j = q^j\left[DBE_j - C\hat{A}\tilde{P}_j\right] \tag{3.38}$$

where $q^j\left[DBE_j - C\hat{A}\tilde{P}_j\right]$ means the filter $\left[DBE_j - C\hat{A}\tilde{P}_j\right]$ with first j terms truncated

Note that the above equation is obtained using (3.30) and (3.35).

Remark 3.3-7:

Equation (3.25) for the SDP can be written as:

$$y_p(t+j|t) = \frac{B}{A}u(t+j-1) + \frac{F_j}{C}x(t) \tag{3.39}$$

or

$$y_p(t+j|t) = y_m(t+j) + \frac{F_j}{C}[y(t) - y_m(t)] \tag{3.40}$$

where y_m is the modelled output and $y(t)$ is the actual (measured) process output.

The above predictor equation has a structure similar to the Kalman filter (Brown, 1983) in the way it assimilate the new information contained in the measurement. In words it can be written as follows:

Updated Prediction = *Model Prediction* + Gain [*Measured Output* - *Model Output*]

where gain is given by $\dfrac{F_j}{C}$.

Remark 3.3-8:

The $y_p(t+j|t)$ in equation (3.25) for SDP can also be interpreted as the sum of known input and uncertainty contributions. The uncertainty (unfiltered), $x(t)$, is projected in future by the factor $\dfrac{F_j}{C}$ which is a function of the employed noise/disturbance model and j.

Figure 3.2 illustrates the uncertainty projection at various frequencies (ω) along the output horizon for a GPC noise model. The *A* polynomial for the noise model is the denominator of the following z-domain process:

$$\text{z - domain transfer function} = \frac{.00768z^{-1} + .02123z^{-2} + .00357z^{-3}}{1 - 1.9031z^{-1} + 1.1514z^{-2} - .2158z^{-3}} \qquad (3.41)$$

Due to the presence of the *A* polynomial in the denominator of the noise model the projected uncertainty amplification factor, $\frac{F_j}{C}$, increases with both j and ω. For ω=0 the amplification factor remains constant at 1 for all values of j. At higher frequencies this factor increases by two order of magnitude for sufficiently large j. If a major proportion of the uncertainty is MPM (Model Plant Mismatch) then the high uncertainty amplification factor may destabilize the predictor. On the other hand a higher amplification factor is desirable for low frequency disturbance rejection provided there is no or low MPM/noise. Inclusion of a *C* polynomial gives a means of decreasing the uncertainty amplification factor. Figure 3.3 which uses $C = \left[1 - 0.9q^{-1}\right]$ and the same GPC noise model of Figure 3.2 demonstrate this fact.

The DMC noise model is a simple integrator (1/Δ) and there is no way in the original formulation to increase (or decrease) the uncertainty amplification factor. Therefore in DMC the residual (i.e. uncertainty) at time = $t+j$ is equal to that at time = t. The generalized noise/disturbance model of equation (3.1) provides three filters to control the uncertainty amplification factor described as follows:

- The *C* polynomial to decrease the uncertainty amplification factor
- The *D* polynomial to increase the uncertainty amplification factor independent of the process model
- The *A* polynomial to increase the uncertainty amplification factor. The increase in the factor is directly proportional to the "slowness" (or the effective time constant) of the process as given by the model (not the actual process).

Figure 3.4 shows that the uncertainty amplification factor for DMC increases from its normal value of unity when the noise model is augmented with a first order D polynomial ($D = \left[1 - 0.9q^{-1}\right]$). In Figure 3.5 uncertainty amplification factors for various noise models are compared at a fixed frequency ($\omega=\pi/2$). The A polynomial for GPC is the denominator of the process transfer function given in equation (3.41), while $C = \left[1 - 0.9q^{-1}\right]$ and $D = \left[1 - 0.8q^{-1}\right]$. GPC gives the highest uncertainty amplification factors whereas the DMC predictor projects the uncertainty very conservatively with a constant uncertainty amplification factor of 1. GPC uncertainty projection is tuned down by the C polynomial and the DMC uncertainty projection is tuned up by the D polynomial.

Remark 3.3-9:

The SDP gives the j-step ahead output prediction as the sum of contributions due to manipulated variable and the residual. Occasionally it is desirable to have the prediction corresponding to a noise model in terms of the one based on a different noise model. In order to derive a useful expression for this situation the following two noise model Diophantine equations will be used:

$$\eta_1 = \frac{C_1}{D_1} = E_{1j} + q^{-j}\frac{F_{1j}}{D_1} \tag{3.42}$$

$$\eta_2 = \frac{C_2}{D_2} = E_{2j} + q^{-j}\frac{F_{2j}}{D_2} \tag{3.43}$$

The j-step ahead predictions based on SDP derived using the above two noise models are denoted by $y_p(t+j|t)_{\eta_1}$ and $y_p(t+j|t)_{\eta_2}$ and are given as follows:

$$y_p(t+j|t)_{\eta_1} = \frac{B}{A}u(t+j-1) + \frac{F_{1j}}{C_1}x(t) \tag{3.44}$$

$$y_p(t+j|t)_{\eta_2} = \frac{B}{A}u(t+j-1) + \frac{F_{2j}}{C_2}x(t) \tag{3.45}$$

Subtraction of (3.44) from (3.45) gives

$$y_p(t+j|t)_{\eta_2} = y_p(t+j|t)_{\eta_1} + \frac{C_1 F_{2j} - C_2 F_{1j}}{C_2 C_1} x(t) \qquad (3.46)$$

or using the Diophantine equations for the two noise models

$$y_p(t+j|t)_{\eta_2} = y_p(t+j|t)_{\eta_1} + \frac{q^j\left\{C_2 D_1 E_{1j} - C_1 D_2 E_{2j}\right\}}{C_2 C_1} x(t) \qquad (3.47)$$

where $q^j\left\{C_2 D_1 E_{1j} - C_1 D_2 E_{2j}\right\}$ represents the filter $\left\{C_2 D_1 E_{1j} - C_1 D_2 E_{2j}\right\}$ with the first j terms truncated.

As an example of this predictor consider the following two noise models and their Diophantine equations:

ARX noise model: $\frac{1}{A} = E_{1j} + q^{-j}\frac{F_{1j}}{A}$ (3.48)

ARARMAX noise model : $\frac{C}{AD} = E_{2j} + q^{-j}\frac{F_{2j}}{AD}$ (3.49)

The ARARMAX model is a generalized equation error structure (Ljung, 1987). The predictor corresponding to ARARMAX in terms of the predictor based on the ARX model is obtained by substituting the numerators and the denominators of the ARX and ARARMAX noise models in (3.46) or (3.47) and is given as:

$$y_p(t+j|t)_{\text{ARARMAX}} = y_p(t+j|t)_{\text{ARX}} + \frac{F_{2j} - CF_{1j}}{C} x(t) \qquad (3.50)$$

or

$$y_p(t+j|t)_{\text{ARARMAX}} = y_p(t+j|t)_{\text{ARX}} + \frac{q^j\left\{CAE_{1j} - ADE_{2j}\right\}}{C} x(t) \qquad (3.51)$$

The predictor (3.51) is identical to the following unified j-step ahead predictor of Soeterboek (1990) used in his Unified Predictive Control (UPC):

$$y_p(t+j|t)_{\text{ARARMAX}} = y_p(t+j|t)_{\text{ARX}} + \frac{K_j A}{C} x(t) \qquad (3.52)$$

note that $K_j A = q^j\left\{CAE_{1j} - ADE_{2j}\right\}$.

Note that the predictor (3.46) or its alternate form (3.47) are very general since they give the output prediction corresponding to any noise model in terms of the prediction based on a different noise model. For example these predictors can be used as an alternative means of deriving and/or interpreting:

- IDMC in terms of classical DMC (Chapter 2)
- EDMC in terms of classical DMC (Chapter 3)
- GPC in terms of DMC
- GPC with T-filter in terms of GPC without T-filter

A brief comparison of UMPC with UPC of Soeterboek (1990 and 1992) is presented in Appendix 3-A.

3.4 The Separated Diophantine Overparameterized Predictor (SDOP)

One of the drawbacks of the LRPC such as GPC is the solution of the Diophantine equations which requires huge computational and storage resources (Kramer, 1991). In this section a new concept of oveparameterized Diophantine equation is presented to overcome this drawback.

The process model can be separated using the following Diophantine expansion:

$$\frac{B}{\Delta^n A} = \widetilde{P}_N + q^{-N} \frac{\overline{P}_N}{\Delta^n A} \tag{3.53}$$

where N can be any integer greater than or equal to j. A useful case results when N is set equal to N_2+1 since $\widetilde{P}_N$ is then of degree N_2 and can be decomposed into past and future components at t+j time instant as follows:

$$\widetilde{P}_N = \widetilde{\widetilde{P}}_{Nj} + q^{-j} \overline{\widetilde{P}}_{Nj} \tag{3.54}$$

Moreover

$$\tilde{\bar{P}}_{Nj} = \tilde{P}_j \tag{3.55}$$

Similarly the noise model can be separated using the following Diophantine expansion:

$$\frac{C}{D\bar{A}\Delta^n} = E_N + q^{-N}\frac{F_N}{D\bar{A}\Delta^n} \tag{3.56}$$

where again E_N, which is of degree N_2, can be decomposed into past and future components at $t+j$ time instant as follows:

$$E_N = \tilde{E}_{Nj} + q^{-j}\overline{E}_{Nj} \tag{3.57}$$

$$\tilde{E}_{Nj} = E_j \tag{3.58}$$

With the above equations and following the procedure used earlier in the SDP derivation the final equation for the overparameterized predictor is given as:

$$\boxed{\begin{aligned} y_p(t+j|t) = \tilde{P}_j\Delta^n u(t+j-1) + \overline{\tilde{\bar{P}}}_{Nj}\Delta^n u(t-1) + \overline{P}_N u^F(t+j-N-1) \\ + \overline{E}_{Nj}e(t) + F_N x^f(t-N+j) \end{aligned}} \tag{3.59}$$

SDOP

Remark 3.4-1:

The SDOP has two parts. The first part consists of the first three terms and it depends exclusively on the plant model. The second part which consists of the last two terms in the SDOP depends only on the employed noise model.

Remark 3.4-2:

The SDOP retains all the advantages associated with SDP mentioned earlier plus it is computationally more efficient than SDP. The reduction in the computational load is achieved by using a single overparameterized Diophantine equation in place of a set of Diophantine equations (viz. one equation for every j) for both process and noise models.

Remark 3.4-3:

For $N=j$, the SDOP reduces to the SDP. This is due to the fact that $\overline{\tilde{\bar{P}}}_{Nj}$ and $\overline{E}_{Nj}$ become zero and $F_N = F_j$ for $N=j$.

Remark 3.4-4:

The computational load in calculating the polynomials $\widetilde{P}_j$, $\overline{\overline{P}}_{Nj}$ and $\overline{E}_{Nj}$ is trivial because these are obtained by simply picking or separating the first j elements from the corresponding degree N_2 polynomials in equations (3.54) and (3.57).

Remark 3.4-5:

The additional variable $e(t)$ can easily be constructed as:

$$e(t) = D\breve{A}\Delta^n x^f(t) \tag{3.60}$$

Remark 3.4-6:

The SDOP is mathematically equivalent to the LDP. The following theorem proves this.

Theorem-2:

The SDOP (equation (3.25)) is mathematically identical to the LDP (equation (3.17)).

Proof:

It is already proven as part of Theorem-1 that $\widetilde{G}_j = \widetilde{P}_j$. Then substituting (3.23) for $x^f(t)$ in (3.59) gives

$$\begin{aligned} y_p(t+j|t) = {} & \widetilde{P}_j\Delta^n u(t+j-1) + \overline{\overline{P}}_{Nj}\Delta^n u(t-1) + \overline{P}_N u^F(t+j-N-1) \\ & + \overline{E}_{Nj}e(t) - F_N\frac{B}{C}u^F(t-N+j-1) + F_N y^f(t-N+j) \end{aligned} \tag{3.61}$$

Changing $u^F(t-1)$ into $\Delta^n u^f(t-1)$ using (3.21) and (3.16)

$$\begin{aligned} y_p(t+j|t) = {} & \widetilde{P}_j\Delta^n u(t+j-1) + \overline{\overline{P}}_{Nj}\Delta^n u(t-1) + \overline{E}_{Nj}e(t) \\ & + \frac{\overline{P}_N C - F_N B}{\Delta^n \breve{A}}\Delta^n u^f(t-N+j-1) + F_N y^f(t-N+j) \end{aligned} \tag{3.62}$$

Multiplying (3.53) throughout by C gives:

$$CB = \Delta^n AC\widetilde{P}_N + q^{-N}\overline{P}_N C \tag{3.63}$$

and multiplying (3.56) throughout by B gives:

$$BC = BD\breve{A}\Delta^n E_N + q^{-N}F_N B \tag{3.64}$$

Equating (3.63) and (3.64) gives

$$q^{-N}\left[\overline{P}_N C - F_N B\right] = \Delta^n\breve{A}\left[DBE_N - \hat{A}C\widetilde{P}_N\right] \tag{3.65}$$

Rearranging

$$\frac{q^{-N}\left[\overline{P}_N C - F_N B\right]}{\Delta^n \breve{A}} = \left[DBE_N - \hat{A}C\widetilde{P}_N\right] = DBE_j + q^{-j} DB\overline{E}_{Nj} - \hat{A}C\widetilde{P}_j - q^{-j}\hat{A}C\overline{\overline{P}}_{Nj} \quad (3.66)$$

$$\begin{aligned}\frac{\overline{P}_N C - F_N B}{\Delta^n \breve{A}}\Delta^n u^f(t - N + j - 1) &= DBE_j \Delta^n u^f(t + j - 1) + DB\overline{E}_{Nj}\Delta^n u^f(t - 1) \\ &\quad - \hat{A}C\widetilde{P}_j \Delta^n u^f(t + j - 1) - \overline{\overline{P}}_{Nj}\Delta^n u(t - 1)\end{aligned} \quad (3.67)$$

Next substituting (3.23) for $e(t)$ gives:

$$\overline{E}_{Nj} e(t) = \frac{\overline{E}_{Nj} D\breve{A}\Delta^n}{C}\left[y(t) - \frac{B}{A}u(t - 1)\right] = \overline{E}_{Nj} D\breve{A}\Delta^n y^f(t) - \overline{E}_{Nj} DB\Delta^n u^f(t - 1) \quad (3.68)$$

Equation (3.56) is used to yield:

$$\begin{aligned}F_N &= q^N\left[C - E_N D\breve{A}\Delta^n\right] \\ F_N y^f(t - N + j) &= q^j\left[C - E_N D\breve{A}\Delta^n\right]y^f(t)\end{aligned} \quad (3.69)$$

$$F_N y^f(t - N + j) = q^j\left[C - E_j D\breve{A}\Delta^n\right]y^f(t) - \overline{E}_{Nj} D\breve{A}\Delta^n y^f(t) \quad (3.70)$$

$$F_N y^f(t - N + j) = F_j y^f(t) - \overline{E}_{Nj} D\breve{A}\Delta^n y^f(t) \text{ because } q^j\left[C - E_j D\breve{A}\Delta^n\right] = F_j \quad (3.71)$$

Inserting $\overline{E}_{Nj} e(t)$, $\frac{\overline{P}_N C - F_N B}{\Delta^n \breve{A}}\Delta^n u^f(t - N + j - 1)$, and $F_N y^f(t - N + j)$ in (3.62) and simplifying yields:

$$y_p(t + j|t) = \widetilde{P}_j \Delta^n u(t + j - 1) + q^j\left[DBE_j - \hat{A}C\widetilde{P}_j\right]\Delta^n u^f(t - 1) + F_j y^f(t) \quad (3.72)$$

Equation (3.35) gives:

$$q^j\left[DBE_j - \hat{A}C\widetilde{P}_j\right] = \frac{C\overline{P}_j - BF_j}{\Delta^n \breve{A}} \quad (3.73)$$

and $\overline{G}_j$ is obtained from (3.36):

$$\frac{C\overline{P}_j - BF_j}{\Delta^n \breve{A}} = \overline{G}_j \quad (3.74)$$

and therefore the SDOP gives the LDP. ◆◆◆

3.5 The Control Law

The purpose of a long-range predictive controller is to find the current (at time = t) control action by considering the effect of current plus future control moves on the output over a future horizon bounded by times $t+N_1$ and $t+N_2$. Time $t+N_1$ is greater than or equal to the earliest time in the future at which the output is affected by the control move at time t, while N_2 is arbitrary but must be greater than N_1.

To date most of the popular long-range predictive controllers use the following type of cost function for the optimal control calculation (Clarke, 1987):

$$J = \sum_{j=N_1}^{N_2} \gamma(j)\left[y_p(t+j|t) - w(t+j)\right]^2 + \sum_{j=N_1}^{N_u} \lambda(j)\left[\Delta^n u(t+j-1)\right]^2 \tag{3.75}$$

where

$\gamma(.)$ = the weighting on tracking error.

$y_p(.)$ = the predicted output values.

$w(.)$= the desired output values or the set-points.

N_1 = the initial output horizon ≥ the earliest future output that is affected by the control move at time t.

N_2 = the final output horizon, the farthest future output that is included in the cost function J.

$\lambda(.)$ = the control weighting sequence.

N_u = the control horizon, the number of future non-zero control moves.

The $y_p(t+j|t)$ in the above cost function is usually given by the LDP, SDP or the SDOP developed above and repeated here for easy reference:

$$y_p(t+j|t) = \widetilde{G}_j \Delta^n u(t+j-1) + \overline{G}_j \Delta^n u^f(t-1) + F_j y^f(t) \tag{3.17}$$

LDP

$$y_p(t+j|t) = \widetilde{P}_j \Delta^n u(t+j-1) + \overline{P}_j u^F(t-1) + F_j x^f(t) \tag{3.25}$$

SDP

$$y_p(t+j|t) = \widetilde{P}_j \Delta^n u(t+j-1) + \overline{\overline{P}}_{Nj} \Delta^n u(t-1) + \overline{P}_N u^F(t+j-N-1) + \overline{E}_{Nj} e(t) + F_N x^f(t-N+j) \tag{3.59}$$

SDOP

The prediction consists of two parts. The first term on the RHS of the predictor is the response due to the future input moves and is known as the *forced response*. The remaining terms on the RHS of the predictor are called the *free-response*. It is the response due to the past inputs $u(.)$ and disturbances up to time t.

The forced response is given by:

$$\begin{aligned} &\text{Forced Response for SDP and SDOP} = \widetilde{P}_j \Delta^n u(t+j-1) \\ &\text{Forced Response for LDP} = \widetilde{G}_j \Delta^n u(t+j-1) \end{aligned} \tag{3.76}$$

where

$$\widetilde{P}_j = \widetilde{G}_j = g_0 + g_1 q^{-1} + \ldots + g_{j-1} q^{-j+1} \tag{3.77}$$

The free response is given by:

$$\text{LDP free response} = f(t+j) = \overline{G}_j \Delta^n u^f(t-1) + F_j y^f(t) \tag{3.78}$$

$$\text{SDP free response} = f(t+j) = \overline{P}_j u^F(t-1) + F_j x^f(t) \tag{3.79}$$

$$\text{SDOP free response} = f(t+j) = \overline{\overline{P}}_{Nj} \Delta^n u(t-1) + \overline{P}_N u^F(t+j-N-1) + \overline{E}_{Nj} \Delta^n e(t) + F_N x^f(t-N+j) \tag{3.80}$$

The control law is given as (McIntosh, 1988):

$$\mathbf{u} = \left[\mathbf{G}^{\mathrm{T}} \Gamma \mathbf{G} + \Lambda\right]^{-1} \mathbf{G}^{\mathrm{T}} \Gamma (\mathbf{w} - \mathbf{f}) \tag{3.81}$$

where

$$\mathbf{u} = \left[\Delta u(t) \quad \Delta u(t+1) \quad \cdots \quad \Delta u(t+Nu-1)\right]^{\mathrm{T}} \tag{3.82}$$

$$\mathbf{w} = \left[w(t+N_1) \quad w(t+N_1+1) \quad \cdots \quad w(t+N_2)\right]^{\mathrm{T}} \tag{3.83}$$

$$\mathbf{f} = \left[f(t+N_1) \quad f(t+N_1+1) \quad \cdots \quad f(t+N_2)\right]^{\mathrm{T}} \tag{3.84}$$

$$\Gamma = \mathrm{diag}[\gamma(N_1) \quad \gamma(N_1+1) \quad \cdots \quad \gamma(N_2)] \tag{3.85}$$

$$\Lambda = \mathrm{diag}[\lambda(1) \quad \lambda(2) \quad \cdots \quad \lambda(N_u)] \tag{3.86}$$

$$\mathbf{G} = \begin{bmatrix} g_{N_1-1} & g_{N_1-2} & \cdots & g_0 & 0 & \cdots & 0 \\ g_{N_1} & g_{N_1-1} & \cdots & g_1 & g_0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & & \ddots & g_0 \\ \vdots & \vdots & \ddots & \ddots & & & g_1 \\ \vdots & \vdots & & \ddots & & & \vdots \\ g_{N_2-1} & g_{N_2-2} & \cdots & \cdots & \cdots & \cdots & g_{N_2-N_u} \end{bmatrix} \tag{3.87}$$

3.6 Enhanced Dynamic Matrix Control (EDMC)

DMC in its various forms is well-established in the chemical industry. However a major drawback of DMC is its slow speed of disturbance rejection. A control horizon (N_u) of one in GPC is often very effective, whereas DMC may require N_u between 3 to 5 to achieve a similar disturbance rejection speed (Clarke, 1991).

In the following formulation a simple algorithm is developed to enhance the disturbance rejection speed of classical DMC. Note that the algorithm is extracted from the generalized noise/disturbance model structure presented earlier.

3.6.1 Classical DMC Predictor using SDP

The measured output model for classical DMC can be written as:

$$y(t) = Hu(t-1) + \frac{1}{\Delta}e(t) = Hu(t-1) + x(t) \tag{3.88}$$

Note this is UMPC with $A = C = D = 1$, $B = H$ and $n = 1$. The residual $x(t)$ at time t and H are given by:

$$H = \sum_{i=1}^{\infty} h_i q^{-i+1} \qquad \text{where the } h_i \text{ are impulse - response coefficients} \tag{3.89}$$

$$x(t) = y(t) - Hu(t-1) = y(t) - \left[\sum_{i=1}^{\infty} h_i q^{-i+1}\right] u(t-1) \tag{3.90}$$

For stable processes the summation can be truncated at *T*, the settling time, giving the following expression for *x*(*t*).

$$x(t) = y(t) - \left[\sum_{i=1}^{T} h_i q^{-i+1}\right] u(t-1) \tag{3.91}$$

The output *y*(*t*) in terms of step-response coefficients is:

$$y(t) = S\Delta u(t-1) + \frac{1}{\Delta} e(t) = S\Delta u(t-1) + x(t) \tag{3.92}$$

where

$$\begin{aligned} S &= \frac{H}{\Delta} = \sum_{i=1}^{\infty} \frac{h_i}{\Delta} q^{-i+1} = \sum_{i=1}^{\infty} s_i q^{-i+1} \quad \text{where } s_i \text{ are step - response coefficients} \\ s_i &= \sum_{k=1}^{i} h_k \qquad \text{and} \quad h_i = s_i - s_{i-1} \quad \text{with } s_0 = 0 \text{ (Seborg et al., 1989)} \end{aligned} \tag{3.93}$$

$$x(t) = y(t) - S\Delta u(t-1) = y(t) - \left[\sum_{i=1}^{\infty} s_i q^{-i+1}\right] \Delta u(t-1) \tag{3.94}$$

Again for stable processes the summation can be truncated at *T* provided that an additional term is included to account for the effect of past inputs beyond the settling time. The following expression for *x*(*t*) is obtained:

$$x(t) = y(t) - \left[\sum_{i=1}^{T} s_i q^{-i+1}\right] \Delta u(t-1) - s_T u(t-T-1) \tag{3.95}$$

The Diophantine equation (3.6) becomes:

$$\frac{1}{\Delta} = E_j + q^{-j} \frac{F_j}{\Delta} = \frac{1-q^{-j}}{\Delta} + q^{-j} \frac{1}{\Delta} \tag{3.96}$$

which implies

$$E_j = \frac{1-q^{-j}}{\Delta} \qquad \text{and} \qquad F_j = 1 \tag{3.97}$$

$$S = \tilde{S}_j + q^{-j}\bar{S}_j \qquad \tilde{S}_j = \sum_{i=1}^{j} s_i q^{-i+1} \qquad \bar{S}_j = \sum_{i=j+1}^{\infty} s_i q^{-i-j+1} \tag{3.98}$$

$$\tilde{P}_j = \tilde{S}_j = \sum_{i=1}^{j} s_i q^{-i+1} \quad \text{and} \quad \frac{\bar{P}_j}{\Delta} = \bar{S}_j = \sum_{i=j+1}^{\infty} s_i q^{-i-j+1}$$

Substituting the above in the SDP equation (3.25) gives:

$$y_p(t+j|t) = \left[\sum_{i=1}^{j} s_i \Delta u(t+j-i)\right] + \left[\sum_{i=j+1}^{\infty} s_i \Delta u(t+j-i)\right] + x(t) \tag{3.99}$$

Equation (3.99) is the untruncated classical DMC predictor (Prett and Garcia, 1988). Truncating the second sum at T gives the following DMC predictor:

$$y_p(t+j|t) = \left[\sum_{i=1}^{j} s_i \Delta u(t+j-i)\right] + \left[\sum_{i=j+1}^{T} s_i \Delta u(t+j-i)\right] + s_T u(t+j-T-1) + x(t) \tag{3.100}$$

Substituting equation in gives the DMC predictor in terms of $y(t)$:

$$y_p(t+j|t) = \left[\sum_{i=1}^{j} s_i \Delta u(t+j-i)\right] + \left[\sum_{i=j+1}^{\infty} \left(s_i - s_{i-j}\right) \Delta u(t-i+j)\right] + y(t) \tag{3.101}$$

Truncating the second sum at T gives the following DMC predictor:

$$\boxed{y_p(t+j|t) = \left[\sum_{i=1}^{j} s_i \Delta u(t+j-i)\right] + \left[\sum_{i=j+1}^{T} \left(s_i - s_{i-j}\right) \Delta u(t-i+j)\right] + y(t)} \tag{3.102}$$

3.6.2 Enhanced DMC (EDMC) Predictor using SDP

The noise model used in DMC is trivial, i.e. a single integrator. The performance of DMC can be significantly improved if a first order ARMA transfer function (or a lead-lag element) is incorporated in its noise model. A denominator polynomial (D) substantially increases the disturbance rejection speed, while a numerator polynomial (C) is useful for noise filtering. These polynomials (or filters) have been previously incorporated into DMC on ad hoc basis, but a rigorous formulation based on noise modelling is lacking. A systematic way of incorporating a first order ARMA noise model in the original DMC formulation without requiring an additional Diophantine equation is presented in the following development.

The DMC measured output model with an added first order ARMA filter is:

$$y(t) = Hu(t-1) + \frac{C}{D\Delta}e(t) = Hu(t-1) + \frac{\left[1-\chi q^{-1}\right]}{\left[1-\delta q^{-1}\right]\left[1-q^{-1}\right]}e(t) \tag{3.103}$$

Note that this is UMPC with $A = 1$, $C = \left[1-\chi q^{-1}\right]$, $D = \left[1-\delta q^{-1}\right]1$, $B = H$ and $n = 1$. In terms of the step response:

$$y(t) = S\Delta u(t-1) + \frac{\left[1-\chi q^{-1}\right]}{\left[1-\delta q^{-1}\right]\left[1-q^{-1}\right]}e(t) \tag{3.104}$$

or

$$y(t) = S\Delta u(t-1) + \left[\frac{\alpha}{\Delta} + \frac{\beta}{\left[1-\delta q^{-1}\right]}\right]e(t) \tag{3.105}$$

where

$$\alpha = \frac{1-\chi}{1-\delta} \qquad \text{and} \qquad \beta = \frac{\chi - \delta}{1 - \delta} \tag{3.106}$$

The Diophantine equation (3.6) becomes:

$$\left[\frac{\alpha}{\Delta} + \frac{\beta}{\left[1-\delta q^{-1}\right]}\right] = \left\{\frac{\alpha\left[1-q^{-j}\right]}{\Delta} + \frac{\beta\left[1-\delta^{j}q^{-j}\right]}{\left[1-\delta q^{-1}\right]}\right\} + q^{-j}\left\{\frac{\alpha\left[1-\delta q^{-1}\right] + \beta\delta^{j}\Delta}{\left[1-\delta q^{-1}\right]\Delta}\right\} \tag{3.107}$$

which implies

$$E_j = \left\{\frac{\alpha\left[1-q^{-j}\right]}{\Delta} + \frac{\beta\left[1-\delta^{j}q^{-j}\right]}{\left[1-\delta q^{-1}\right]}\right\} \qquad \text{and} \qquad F_j = \alpha\left[1-\delta q^{-1}\right] + \beta\delta^{j}\Delta \tag{3.108}$$

Substituting the above expression for F_j in the SDP equation and rearranging gives the following :

$$\begin{aligned} y_p(t+j|t) = {} & \left[\sum_{i=1}^{j} s_i \Delta u(t+j-i)\right] + \left[\sum_{i=j+1}^{\infty} s_i \Delta u(t+j-i)\right] + x(t) \\ & + \frac{(\delta-\chi)(1-\delta^{j})}{(1-\delta)}\Delta x^{f}(t) \end{aligned} \tag{3.109}$$

where

$$x^f(t) = \chi x^f(t-1) + x(t) \tag{3.110}$$

Truncating the second sum at T gives the following EDMC predictor:

$$\boxed{\begin{aligned} y_p(t+j|t) = \left[\sum_{j=1}^{j} s_i \Delta u(t+j-i)\right] + \left[\sum_{i=j+1}^{T} s_i \Delta u(t+j-i)\right] + s_T u(t+j-T-1) + x(t) \\ + \frac{(\delta-\chi)(1-\delta^j)}{(1-\delta)} \Delta x^f(t) \\ (3.111) \\ \text{EDMC} \end{aligned}}$$

The above predictor for EDMC can be expressed in terms of the original DMC predictor by:

$$\langle y_p(t+j|t) \rangle_{IDMC} = \langle y_p(t+j|t) \rangle_{DMC} + \frac{(\delta-\chi)(1-\delta^j)}{(1-\delta)} \Delta x^f(t) \tag{3.112}$$

Note that the separate parameterization in SDP makes it possible to include a first order lead-lag (ARMA) filter in a much easier way than the inclusion of only a lag filter in IDMC of Chapter 2 using the LDP. Moreover for $\chi = 0$ the EDMC predictor reduces to the IDMC predictor and for $\chi=\delta=0$ the EDMC predictor reduces to the classical DMC predictor.

3.7 Disturbance Horizon

In most LRPC's the noise model is specified arbitrarily as a means of tuning the disturbance response of the controller. Nevertheless in the presence of unmodelled dynamics (MPM) this ad hoc noise model specification can destabilize the controller. It is also well known that the DMC noise model (i.e. the Random Walk model or a single integrator) gives a robust controller with offset-free response but the disturbance rejection speed is quite low. This is due to the absence of suitable tuning polynomials in the noise model. Thus the DMC noise model can be used as a reference for a conservative but robust controller (see also Remark 3.3.7 and 8).

As mentioned earlier, the separation of the manipulated variable and the residual terms in the SDP offers certain advantages. One of the advantages is that more than one noise/disturbance model can be employed for the output prediction. For example a *primary* noise model can be used to predict the earlier part of the output horizon and a *secondary* model can be employed for the later part. The portion of the output horizon which uses the primary noise model is defined as the "*Disturbance Horizon*". A performance-oriented noise model may be specified for this earlier part of the output horizon, then for the later part a more conservative noise model can be adopted. The term "Disturbance Horizon" is suggested for this earlier part of the output horizon. The most logical choice for a sufficiently robust noise model for use in the latter part of the prediction horizon is the DMC noise model. A combination of noise models over the output prediction horizon can result in a robust controller, i.e. one that will perform well in the presence of MPM.

The SDP is given as:

$$y_p(t+j|t) = \widetilde{P}_j \Delta^n u(t+j-1) + \overline{P}_j u^F(t-1) + F_j x^f(t) \tag{3.25}$$

The free response is given as:

$$\begin{aligned} \text{free response} = f(t+j) &= \overline{P}_j u^F(t-1) \quad + \quad F_j x^f(t) \\ &= \underset{\substack{\text{input}\\ \text{contribution}}}{\overline{P}_j u^F(t-1)} \quad + \quad \underset{\substack{\text{residual}\\ \text{contribution}}}{\frac{F_j}{C} x(t)} \end{aligned} \tag{3.113}$$

The residual $x(t)$ is the output equivalent of the lumped uncertainty (disturbance, noise or MPM) at time t. The projected contribution of $x(t)$ to the output prediction is a function of the filter $\frac{F_j}{C}$ which is exclusively in terms of the noise model.

For a general performance-oriented noise model the filter $\frac{F_j}{C}$ is a complicated function of j and the noise model . However for the DMC noise model $\frac{F_j}{C}=1$. (see Figures 3.2-3.5).

One simple implementation of the disturbance horizon is as follows:

$$\begin{aligned} y_p(t+j|t) &= \widetilde{P}_j\Delta^n u(t+j-1) + \overline{P}_j u^F(t-1) + F_j x^f(t) \quad && \text{for } j = N_1 \text{ to } N_d \\ y_p(t+j|t) &= \widetilde{P}_j\Delta^n u(t+j-1) + \overline{P}_j u^F(t-1) + x(t) && \text{for } j = N_{d+1} \text{ to } N_2 \end{aligned} \qquad (3.114)$$

where N_d is the disturbance horizon. A plot of $\frac{F_j}{C}$ vs. j, for a fixed frequency of $\omega=\pi/2$, for this implementation is illustrated in Figure 3.6. In this case the primary noise model is the GPC without T-filter corresponding to the process transfer function given in equation (3.41) and the secondary noise model is DMC. The output horizon (N_2) is 20 and the disturbance horizon (N_d) is 17. In Figure 3.7, which uses the same process model and output horizon as Figure 3.6, DMC is used the primary noise model over a disturbance horizon of 10 while GPC is used as the secondary noise model for the rest of the output horizon.

Another option is to set the uncertainty contribution to all of the output prediction for $j>N_d$ equal to the uncertainty contribution corresponding to $j=N_d$ as given in the following:

$$\begin{aligned} y_p(t+j|t) &= \widetilde{P}_j\Delta^n u(t+j-1) + \overline{P}_j u^F(t-1) + F_j x^f(t) \quad && \text{for } j = N_1 \text{ to } N_d \\ y_p(t+j|t) &= \widetilde{P}_j\Delta^n u(t+j-1) + \overline{P}_j u^F(t-1) + F_{N_d} x(t) && \text{for } j = N_{d+1} \text{ to } N_2 \end{aligned} \qquad (3.115)$$

This strategy is depicted in Figure 3.8 which uses the GPC noise model, for residual projection, for j = 1 to 8. Then for the rest of output horizon (i.e. j = 9 to 20), the projection of uncertainty is fixed at its value corresponding to j = 8. The for this example is the same used in Figures 3.6 and 7.

An even more UMPC general implementation uses two fully specified noise models with the following Diophantine equations:

$$\begin{aligned} \frac{C}{D\breve{A}\Delta^n} &= E_j + q^{-j}\frac{F_j}{D\breve{A}\Delta^n} \quad && \text{for } j = N_1 \text{ to } N_d \\ \frac{C'}{D'\breve{A}'\Delta^{n'}} &= E'_j + q^{-j}\frac{F'_j}{D'\breve{A}'\Delta^{n'}} && \text{for } j = N_{d+1} \text{ to } N_2 \end{aligned} \qquad (3.116)$$

with these noise models the predictions are given as:

$$\begin{aligned} y_p(t+j|t) &= \widetilde{P}_j \Delta^n u(t+j-1) + \overline{P}_j u^F(t-1) + F_j x^f(t) \quad \text{for } j = N_1 \text{ to } N_d \\ y_p(t+j|t) &= \widetilde{P}_j \Delta^n u(t+j-1) + \overline{P}_j u^F(t-1) + F'_j x^{f'}(t) \quad \text{for } j = N_{d+1} \text{ to } N_2 \end{aligned} \tag{3.117}$$

where

$$x^f(t) = \frac{x(t)}{C} \qquad \text{and} \qquad x^{f'}(t) = \frac{x(t)}{C'}$$

3.8 σ-Weighting

The UMPC formulation discussed above incorporates two (or more) noise models which are used over a specified discrete part of the prediction horizon. A simple way to incorporate multiple noise models with non-discrete (continuous) proportions in the control law is to include a weighted combination of the tracking errors corresponding to both noise models for the entire prediction horizon in the cost function. This can be achieved by defining the following matrices and vectors:

$$\mathbf{G}^+ = \begin{bmatrix} \mathbf{G} \\ \mathbf{G} \end{bmatrix} \qquad \Gamma^+ = \begin{bmatrix} (1-\sigma)\Gamma & 0 \\ 0 & \sigma\Gamma \end{bmatrix} \qquad \mathbf{w}^+ = \begin{bmatrix} \mathbf{w} \\ \mathbf{w} \end{bmatrix} \qquad \mathbf{f}^+ = \begin{bmatrix} \mathbf{f} \\ \mathbf{f}' \end{bmatrix} \tag{3.118}$$

where $\mathbf{f}$ and $\mathbf{f}'$ are free responses corresponding to the first and the second noise models respectively. The control law (3.81) then becomes:

$$\mathbf{u} = \left[\mathbf{G}^{+T}\Gamma^+\mathbf{G}^+ + \Lambda\right]^{-1}\mathbf{G}^{+T}\Gamma^+(\mathbf{w}^+ - \mathbf{f}^+) \tag{3.119}$$

This can be simplified as follows:

$$\mathbf{G}^{+T}\Gamma^+\mathbf{G}^+ = \begin{bmatrix} \mathbf{G}^T & \mathbf{G}^T \end{bmatrix} \begin{bmatrix} (1-\sigma)\Gamma & 0 \\ 0 & \sigma\Gamma \end{bmatrix} \begin{bmatrix} \mathbf{G} \\ \mathbf{G} \end{bmatrix} = \mathbf{G}^T(1-\sigma)\Gamma\mathbf{G} + \mathbf{G}^T\sigma\Gamma\mathbf{G} = \mathbf{G}^T\Gamma\mathbf{G} \tag{3.120}$$

$$\begin{aligned} \mathbf{G}^{+T}\Gamma^+(\mathbf{w}^+ - \mathbf{f}^+) &= \begin{bmatrix} \mathbf{G}^T & \mathbf{G}^T \end{bmatrix} \begin{bmatrix} (1-\sigma)\Gamma & 0 \\ 0 & \sigma\Gamma \end{bmatrix} \begin{bmatrix} \mathbf{w} - \mathbf{f} \\ \mathbf{w} - \mathbf{f}' \end{bmatrix} \\ &= \mathbf{G}^T(1-\sigma)\Gamma(\mathbf{w} - \mathbf{f}) + \mathbf{G}^T\sigma\Gamma(\mathbf{w} - \mathbf{f}') \qquad (3.121) \\ &= \mathbf{G}^T\Gamma(\mathbf{w} - \mathbf{f}_{av}) \qquad \text{where} \qquad \mathbf{f}_{av} = (1-\sigma)\mathbf{f} + \sigma\mathbf{f}' \end{aligned}$$

$$\boxed{\mathbf{u} = \left[\mathbf{G}^T\Gamma\mathbf{G} + \Lambda\right]^{-1}\mathbf{G}^T\Gamma(\mathbf{w} - \mathbf{f}_{av})} \tag{3.122}$$

The control law in (3.122) is the same control law derived previously except the free response is a weighted combination (average) of the free responses obtained using two different noise models. Note that when $\sigma = 0$, $\mathbf{f}_{av} = \mathbf{f}$ and for $\sigma = 1$, $\mathbf{f}_{av} = \mathbf{f}'$. Note also that $0 \leq \sigma \leq 1$ can be changed on line at any time step without having to recalculate any of the Diophantine equations which is ideal for "dynamic tuning" of the controller to accommodate changes in the noise, disturbances and/or MPM. Moreover when the two noise model are identical then (3.122) becomes the original control law given in (3.81).

In scalar form the average free response is given as:

$$f_{av}(t+j) = \overline{P}_j u^F(t-1) + (1-\sigma)F_j x^f(t) + \sigma F_j' x'^f(t) \qquad (3.123)$$

Figure 3.9 describes the interpolating effect of σ-weighting on the uncertainty projection for the noise models of GPC and DMC.

3.9 Simulation Examples

Simulations were designed to analyze and compare the above developed disturbance rejection techniques. In particular these examples are based on the following propositions:

- A DMC noise model gives very slow disturbance rejection.
- EDMC gives faster (stronger) disturbance rejection.
- A disturbance horizon can be used to tune the controller. Use of a DMC noise model for predictions greater than N_d steps ahead can stabilize the system. Multiple noise models are another way of achieving robust and fast disturbance rejection. The following combinations are useful:
  - ⇒ GPC noise model followed by DMC noise model
  - ⇒ DMC noise model followed by GPC noise model
  - ⇒ The Diophantine expansion for GPC noise model is limited beyond a certain point in the output horizon.
- A weighted average of free responses corresponding to GPC and DMC noise models provides a continuous (non-discrete) way of selecting any desired speed of disturbance rejection between the GPC and the DMC responses.

All of the simulations in this example are based on the z-domain 3rd order process given in equation (3.41) repeated in the following for reference

$$\text{z-domain transfer function} = \frac{.00768z^{-1} + .02123z^{-2} + .00357z^{-3}}{1 - 1.9031z^{-1} + 1.1514z^{-2} - .2158z^{-3}} \tag{3.41}$$

The poles of the z-domain transfer function are 0.8187, 0.7165 and 0.3679, while the zeros are at -2.586 and -0.1798. Note that one of the zeros is outside the unit circle making the discretized process nonminimum phase (NMP). The gain of the process is 1. All simulations are carried out for a period of 150 time instants with a setpoint change of unity at time = 0 and a sustained step disturbance of magnitude 0.5 starting at time = 50.

3.9.1 Example-1: DMC and EDMC Performance Comparison

A step response formulation was used for both DMC and EDMC. A step response vector of length 40 is obtained using the transfer function of the plant (3.41). For N_2=30, N_1=1, N_u=1 and λ=0 the classical DMC gives very slow disturbance rejection as is seen in Figure 3.10. EDMC speeds up the disturbance rejection performance by using $D = [1 - 0.9q^{-1}]$ with the classical DMC. Control signal oscillations are reduced when a first order C polynomial ($C = [1 - 0.5q^{-1}]$) is added to the noise model. The role of C in presence of highly noisy measurement data is demonstrated in Figure 3.11 where a C polynomial helps reduce the control signal oscillations when the measurement is corrupted by a white noise with a standard deviation of 0.5. Note that the noise is added directly to the process output and hence is the same in both parts of the top plot in Figure 3.11.

3.9.2 Example-2: Demonstration of Disturbance Horizon as a Tuning Parameter

The disturbance horizon provides a means to incorporate two (or more) noise/disturbance models into the control law design. This example demonstrates the following scenarios:

a) *A GPC noise model followed by the DMC noise model*:

This is given Figure 3.12 which shows that sequential use of GPC and DMC noise models in the prediction horizon give a closed loop response faster than DMC and slower than GPC. The noise model is changed at N_d=17.

b) *The DMC noise model followed by a GPC noise model*:

In Figure 3.13 the order of the noise models is reversed. Here the DMC noise model is used for the first 10 intervals (i.e. N_d=10) and subsequently a GPC noise model is employed. Again the response falls in between the responses corresponding to DMC and GPC. However, this approach is probably more sensitive to high frequency MPM than that used in part a.

c) *Limiting the Expansion of the GPC noise model up to a disturbance horizon* :

In this simulation the Diophantine expansion for $j=N_d=8$ is also employed for $j>N_d$ i.e. the contribution of the residual to the prediction of the process output is constant for $j \geq N_d=8$. Figure 3.14 shows that the disturbance rejection speed for this case is slower than that of GPC but faster than DMC.

3.9.3 Example-3: Multiple Noise Models using σ-Weighting

A weighted combination of free responses corresponding to GPC and DMC can be incorporated into the control algorithm using σ-weighting. This is demonstrated in this example which consists of the following simulations:

a) *A weighted Combination of GPC and DMC noise models*:

Figure 3.15 demonstrates the effect of σ-weighting. Here the primary and the secondary noise models are GPC and DMC respectively. Obviously $\sigma = 0$ is GPC and $\sigma = 1$ is DMC. A σ-weighting of 0.3 moves the GPC response a little towards DMC.

b) *σ-weighting in presence of Model Plant Mismatch*:

In order to demonstrate the role of σ-weighting in presence of unmodeled dynamics the following reduced order model is employed with a GPC noise model.

$$\text{Reduce Order Model} = \frac{0.08z^{-1}}{1-0.91z^{-1}} \tag{3.124}$$

Figure 3.16 shows that the close-loop response is unstable for GPC noise model. However if a σ-weighting of 0.5 is employed with DMC as the secondary noise model a stable response with reasonably fast disturbance rejection speed is obtained. Similar results were obtained in a number of other simulations.

3.10 Conclusions

The two new predictors (i.e. SDP and SDOP) proposed in this chapter facilitate the independent manipulation of the contribution of the uncertainty term to the optimal long range output predictor. An immediate result of the separated formulation of the LRPC predictor is the concept of a disturbance horizon which enables the use of multiple noise models in the control law. A further benefit of SDP permits the use of a weighted sum of multiple noise/disturbance model contributions to the long range free response. This methodology is named σ-weighting. The specific results and conclusions are summarized as follows:

- The Separated Diophantine Predictor (SDP) is the basis for independent manipulation of the uncertainty contribution to the future output predictions
- The SDP-based LRPC formulation is direct, simple and straightforward when compared to the classical LDP-based LRPC algorithm. In addition the SDP formulation is easier to program.
- The SDP-based LRPC formulation provides better insight for theoretical analysis because the uncertainty terms are decoupled from the process terms.
- The Separated Diophantine Overparameterized Predictor (SDOP) offers substantial computational savings while retaining the benefits of SDP-based LRPC.
- Both the SDP and SDOP are optimal predictors and are mathematically identical to the classical LDP.
- SDP readily gives the actual classical DMC predictor as a special case of the generalized noise model as employed in this study.

- An Enhanced form of DMC (EDMC) is obtained by simple derivation using the SDP. The EDMC gives faster disturbance rejection than the classical DMC. In addition EDMC provides a lead/lag observer filter for robustness and noise attenuation.
- The newly introduced notion of Disturbance Horizon offers a systematic way of incorporating more than one noise/disturbance models in the LRPC design. It may be used to tune/detune the controller on-line.
- The σ- weighting concept is another contribution of the present study. It allows the use of multiple noise/disturbance models in the free response predictions in any arbitrary proportions.

References:

Brown, R.G., "Introduction to Random Signal Analysis and Kalman Filtering", pp. 196, John Wiley & Sons, 1983.

Clarke, D.W.," Adaptive Generalized Predictive Control ", Proceedings of the Fourth International Conference on Process Control (CPCIV), Editors Y. Arkun and W. H. Ray, Padre Island, TX, 1991.

Clarke, D.W., Mohtadi, C. and Tuffs, P.S., " Generalized Predictive Control - Part I the Basic Algorithm ", Automatica, Vol. 23, No. 2, pp.137-148, 1987a.

Clarke, D.W., Mohtadi, C. and Tuffs, P.S., " Generalized Predictive Control - Part II Extensions and Interpretations ", Automatica, Vol. 23, No. 2, pp. 149-160, 1987b.

Cutler, C.R., "Dynamic Matrix Control - A Computer Control Algorithm ", AIChE National Meeting, Houston, TX, 1979.

Kramer, K. and Unbehauen, H., "Predictive Adaptive Control - Comparison of Main Algorithms", Proceedings of the First European Control Conference, pp. 327-332, 1991.

Ljung, L., " System Identification: Theory for the User ", Prentice Hall, Englewood Cliffs, New Jersey, 1987.

McIntosh, A.R., "Performance and Tuning of Adaptive Generalized Predictive Control ", M.Sc. thesis, University of Alberta, 1988.

Mohtadi, C., "Studies in Advanced Self-Tuning", D.Phil thesis, Oxford University, 1986.

Prett, D.M., and Garcia, C.E., "Fundamental Process Control", Betterworth Publishers, 1988.

Seborg, D.E., Edgar, T.F. and Mellichamp, D.A., "Process Dynamics and Control", pp. 652, John Wiley & Sons, 1989.

Soeterboek, A.R.M., "Predictive Control — A Unified Approach", Ph.D. thesis, Delft University of Technology, Netherlands, 1990.

Soeterboek, A.R.M., "Predictive Control — A Unified Approach", Prentice Hall International Series in Systems and Control Engineering, 1992.

Appendix 3-A

Comparison of UMPC with the Unified Predictive Control (UPC) of Soeterboek (1990)

The key elements of UPC, with minor nomenclature changes, are summarized below:

The Measured output model:

$$y(t) = \frac{q^{-d}B}{A}u(t-1) + \frac{C}{DA}e(t)$$

Note that this is an ARARMAX model (Ljung, 1987) and has an equation error structure.

Diophantine equations:

$$\frac{C}{DA} = E_j + q^{-j}\frac{F_j}{DA}$$

$$\frac{1}{A} = E'_j + q^{-j}\frac{F'_j}{A}$$

$$CE'_j = M_j + q^{-j-n_D}N_j$$ where n_D is the degree of D

$$\frac{F_j}{A} = Q_j + q^{-n_D}\frac{R_j}{A}$$

$$\frac{F'_j C}{A} = Q'_j + q^{-n_D}\frac{R'_j}{A}$$

$$\frac{1}{N\Delta^n} = E_{j-N_u} + q^{-j+N_u}\frac{F_{j-N_u}}{N\Delta^n}$$ where N is any arbitrary polynomial

The Unified Predictor:

$$y_p(t+j|t) = \tilde{G}_j u(t+j-d-1) + \overline{G}_j u(t-1) + F_j y(t) + \frac{K_j A}{C} x(t)$$

or

$$y_p(t+j|t)_{ARARMAX} = y_p(t+j|t)_{ARX} + \frac{K_j A}{C} x(t)$$

The Cost Function:

$$J = \sum_{j=N_1}^{N_2} \left[P y_p(t+j|t) - R w(t+j) \right]^2 + \lambda \sum_{j=1}^{N_2} \left[\frac{Q_n}{Q_d} u(t+j-1) \right]^2$$

with constraints:

$$N \Delta^n u(t+j-1) = 0 \qquad \text{for} \quad 1 \le N_u < j \le N_2$$

where P, R, Q_n, Q_d and N are polynomials in q^{-1}.

The following are the main differences between UMPC and UPC

- UPC uses an equation error model (the ARARMAX model) whereas, UMPC is based on a more general model which combines the equation error and the output error structures into one unified formulation.

- UPC does not include the integrators in the basic predictor formulation. They are accommodated later in the control law derivation which requires an additional Diophantine equation. UMPC on the other hand incorporates the integrators in the noise model explicitly which does not an additional Diophantine equation.

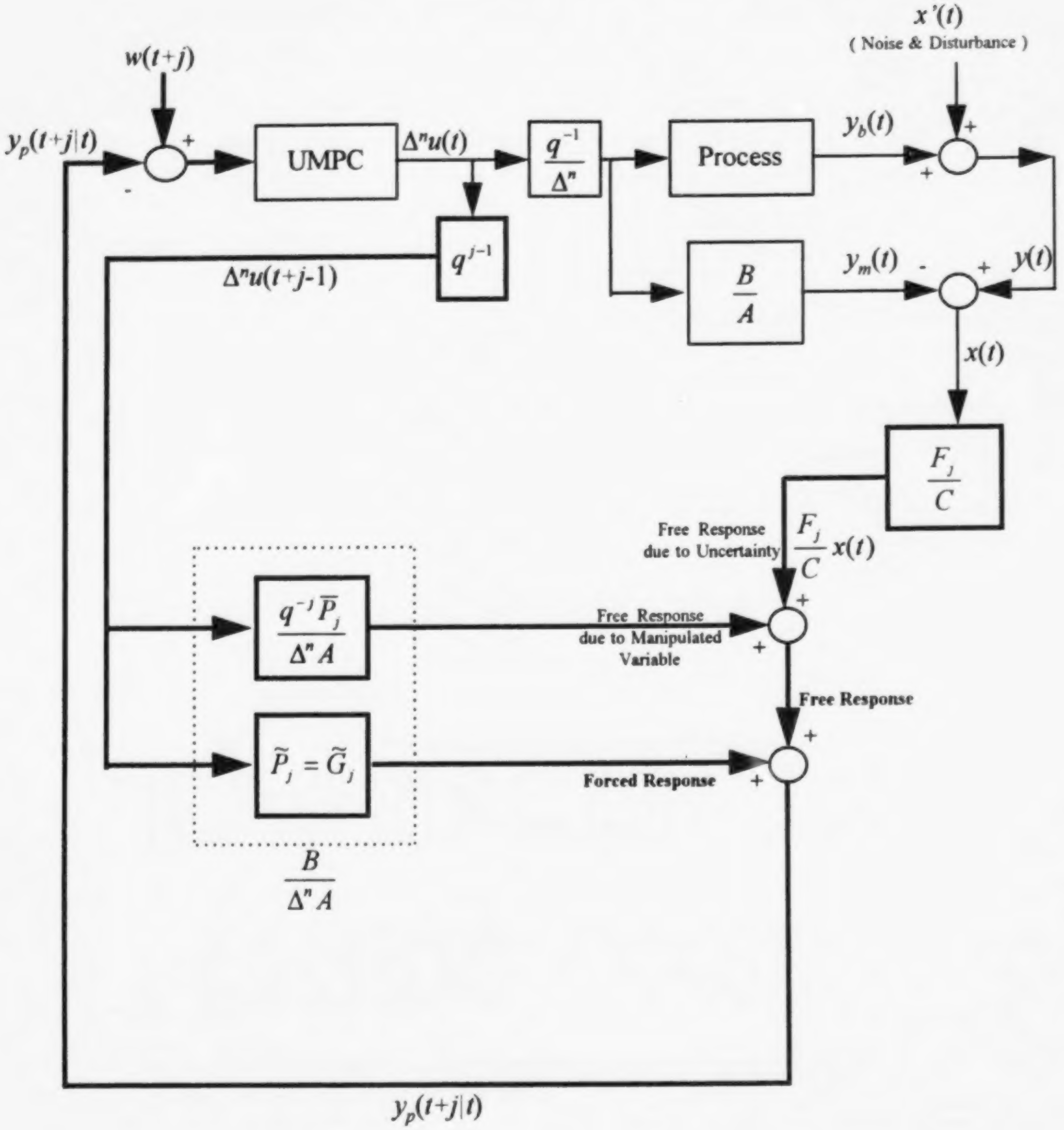
- The unified predictor of UPC needs solution of five Diophantine equations, while the SDP for UMPC requires only two Diophantine expansions.

- The unified predictor of UPC is the sum of a predictor for an ARX model plus an additional term to account for C and D. Since the ARX structure itself includes a noise model, this predictor does not isolate the noise contribution to the output prediction as a separate term. The SDP for UMPC gives an explicit separation of the process and noise contributions to the prediction. Moreover the SDP can also be put in a form such that the predictions associated with any one noise model can be expressed as sum of the prediction based on another noise model and a term that accounts for the difference between the two noise models. Consequently the unified predictor of UPC is a special case of SDP (see Remark 3.3-9).

- UPC incorporates dynamic control weighting while UMPC does not. The importance of dynamic control weighting in LRPC's has not been demonstrated and it is not recommended for tuning (McIntosh, 1988 and Mohtadi, 1986). Moreover an additional Diophantine equation $\left(\frac{Q_n}{Q_d} = \Phi_j + q^{-j}\frac{\Omega_j}{Q_d}\right)$ has to be solved. The absence of the dynamic control weighting greatly simplifies the UMPC formulation.

- UMPC formulation is much simpler and straightforward compared to UPC formulation which involves many intermediate variables such as u^* and y^*.

- UMPC has following features:
    - Computationally cheaper overparameterized predictors
    - Multiple noise models
    - Steady state weighting
    - Sparse output and control horizons
    - Multiple control horizons
    - Recursive formulation


x'(t)
(Noise & Disturbance)
w(t+j)
y_p(t+j|t)
+
-
UMPC
Δ^n u(t)
q^-1/Δ^n
Process
y_b(t)
+
+
q^(j-1)
Δ^n u(t+j-1)
B/A
y_m(t)
-
+
y(t)
x(t)
F_j/C
Free Response due to Uncertainty
F_j/C x(t)
+
q^-j P̄_j / Δ^n A
Free Response due to Manipulated Variable
+
Free Response
+
P̃_j = G̃_j
Forced Response
+
B / Δ^n A
y_p(t+j|t)

Notes:

1) The filters and signals with the index j are vectors and are drawn with thicker lines; j varies from 1 to N_2.

2) The UMPC block does the calculations (optimization) necessary to produce the control vector that minimizes the predicted control errors.

Figure 3.1 Unified model predictive control (UMPC) with separated Diophantine predictor (SDP)

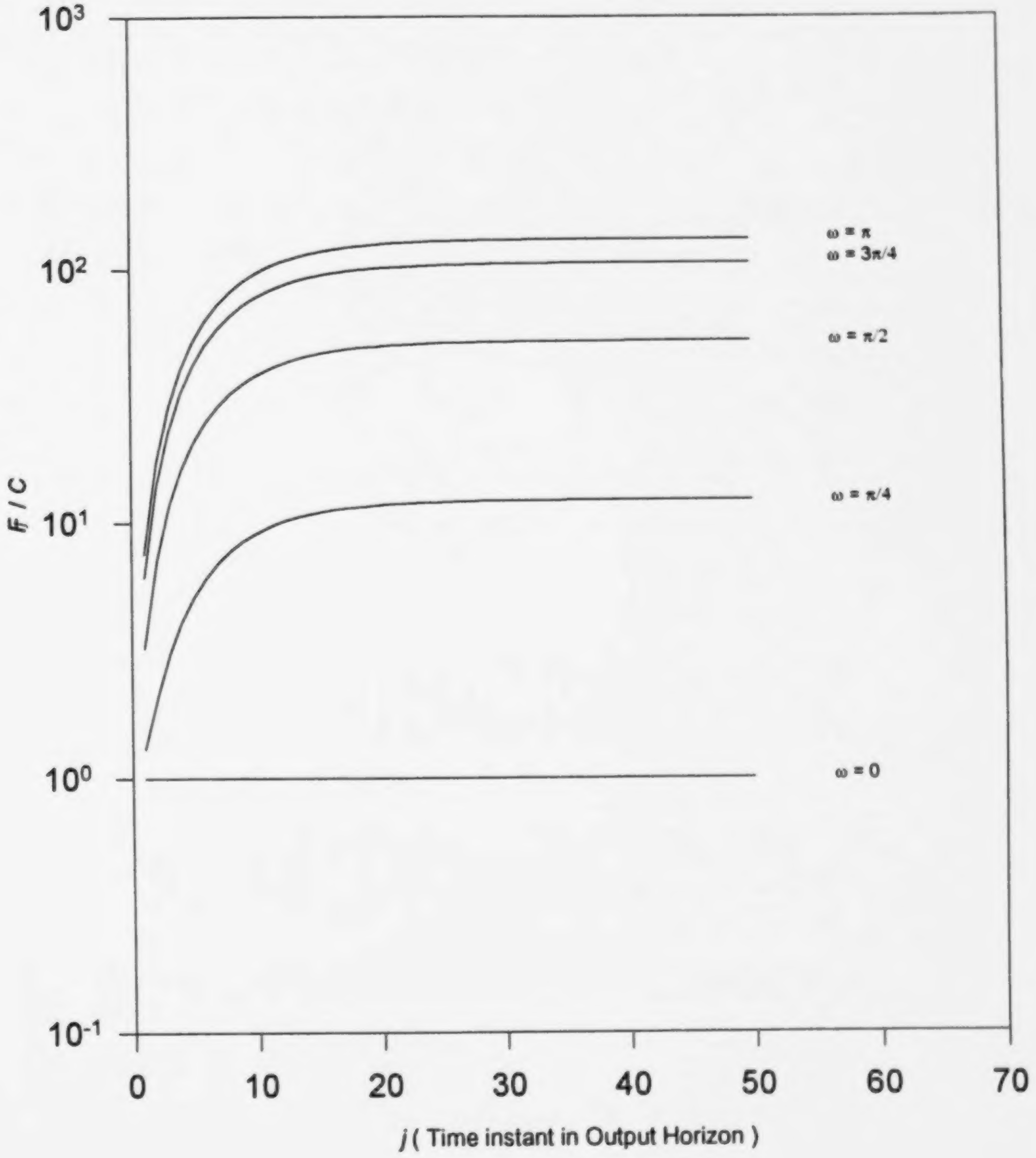


Figure 3.2 Uncertainty projection at various frequencies for GPC noise model

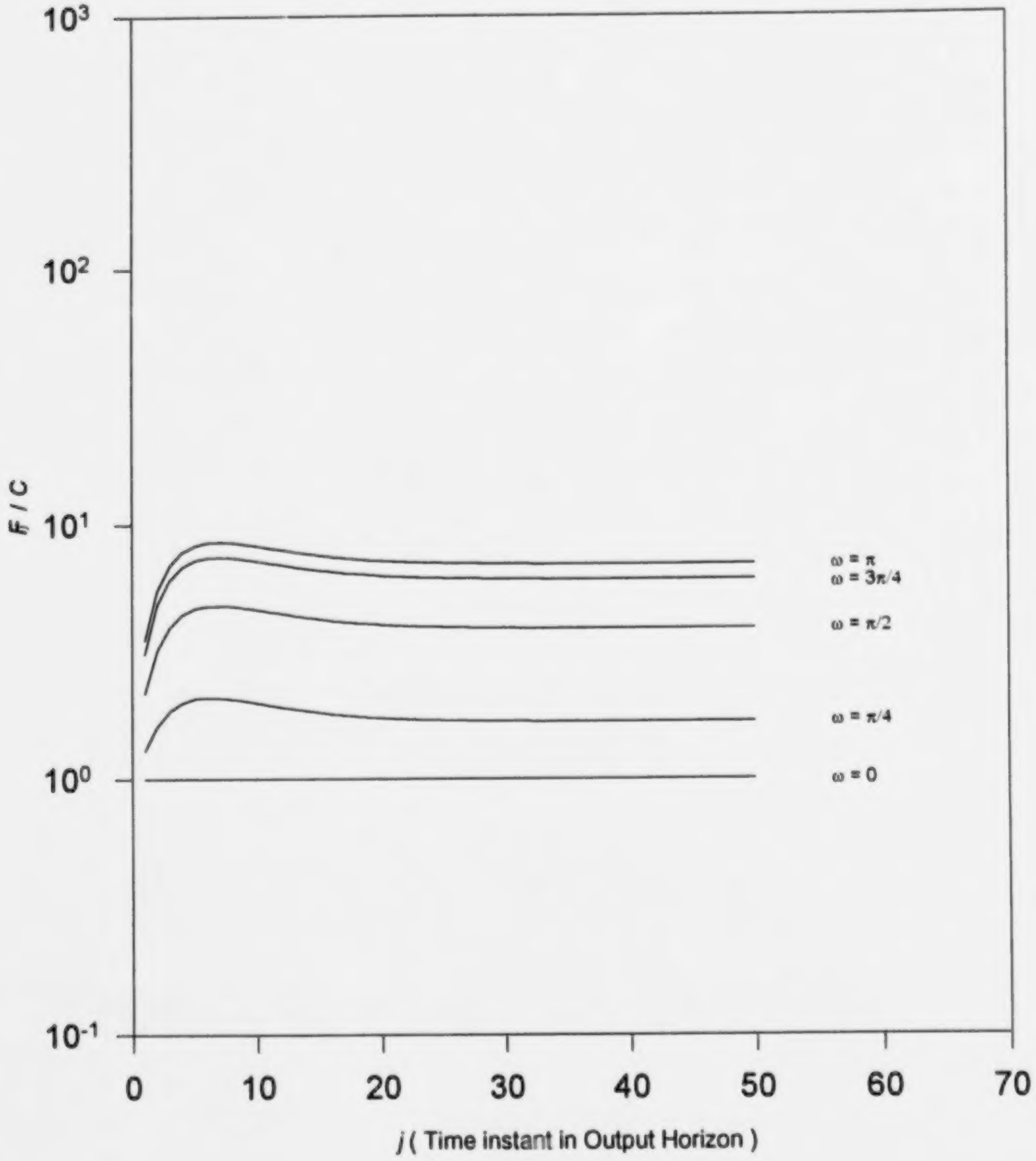


Figure 3.3 Uncertainty projection at various frequencies for GPC with T-filter

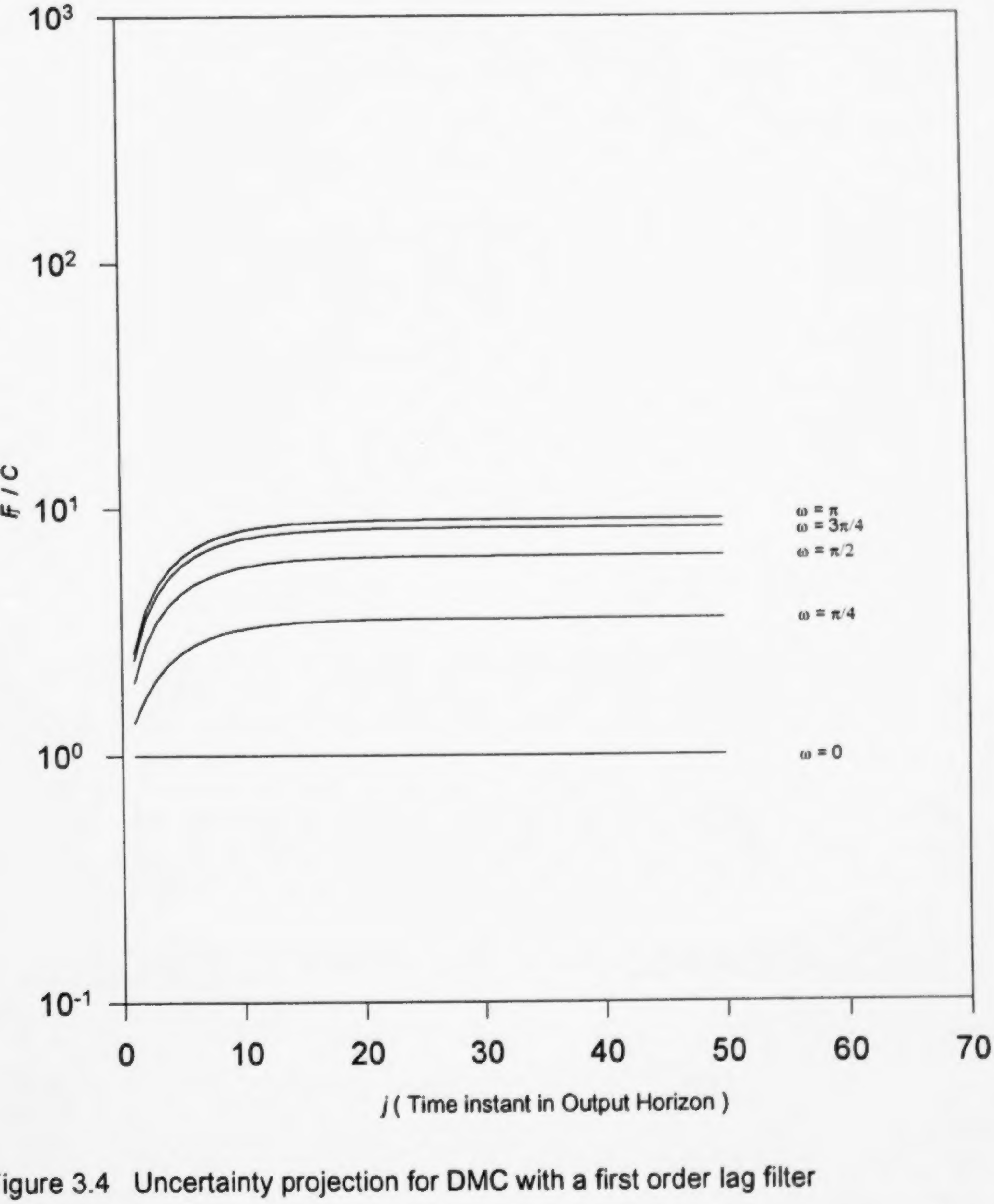


Figure 3.4 Uncertainty projection for DMC with a first order lag filter

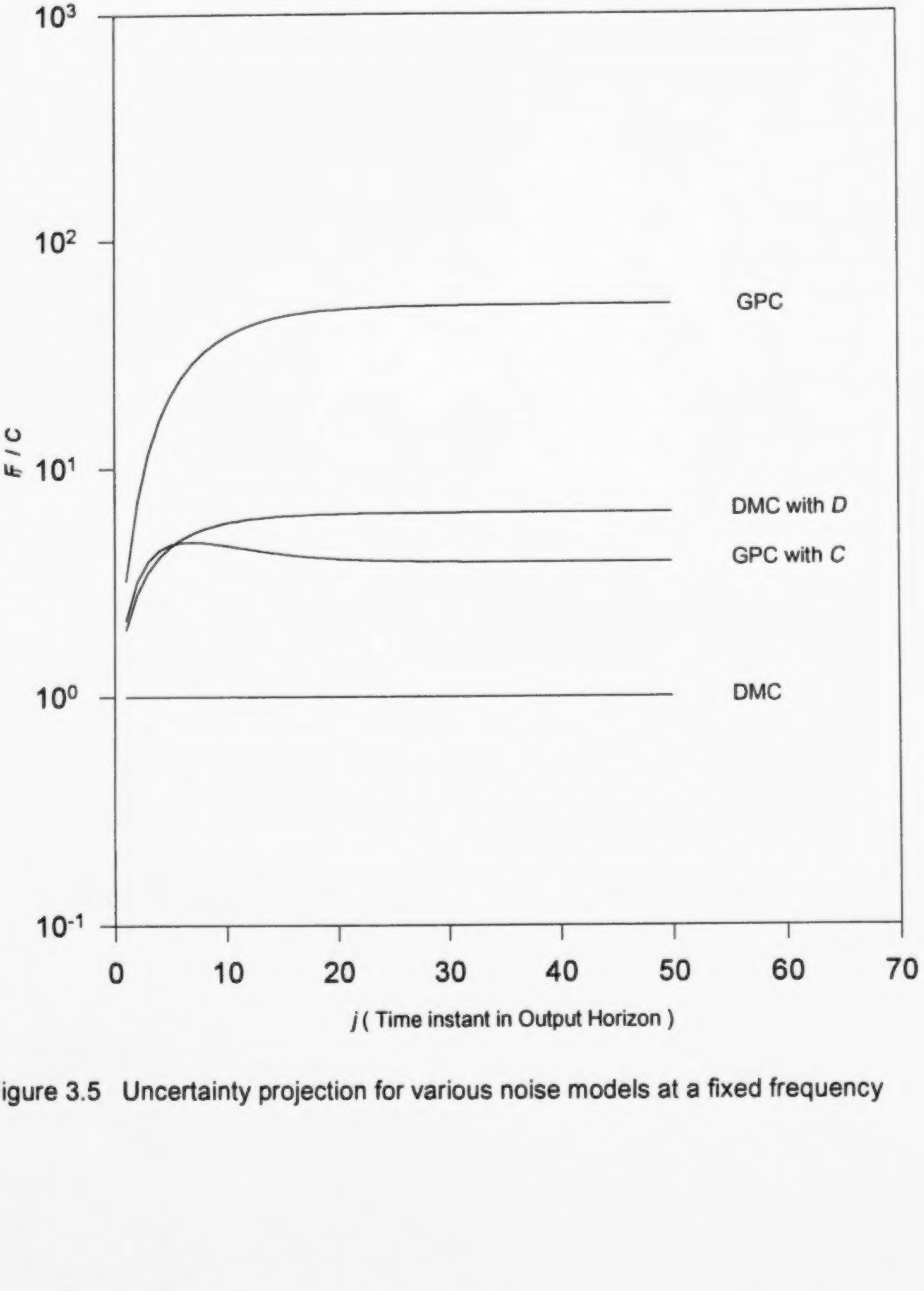


Figure 3.5 Uncertainty projection for various noise models at a fixed frequency

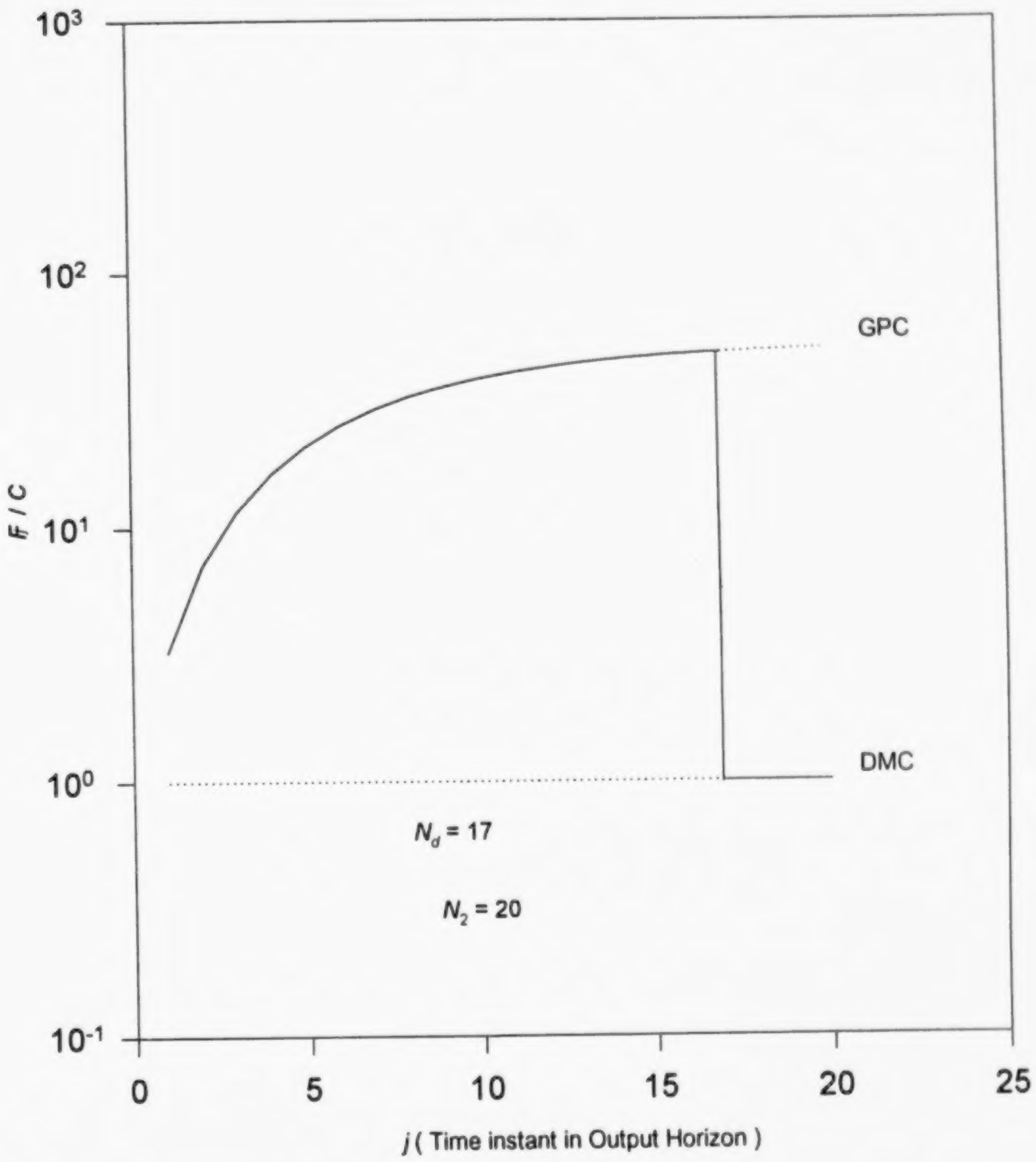


Figure 3.6 Uncertainty projection using disturbance horizon for GPC followed by DMC

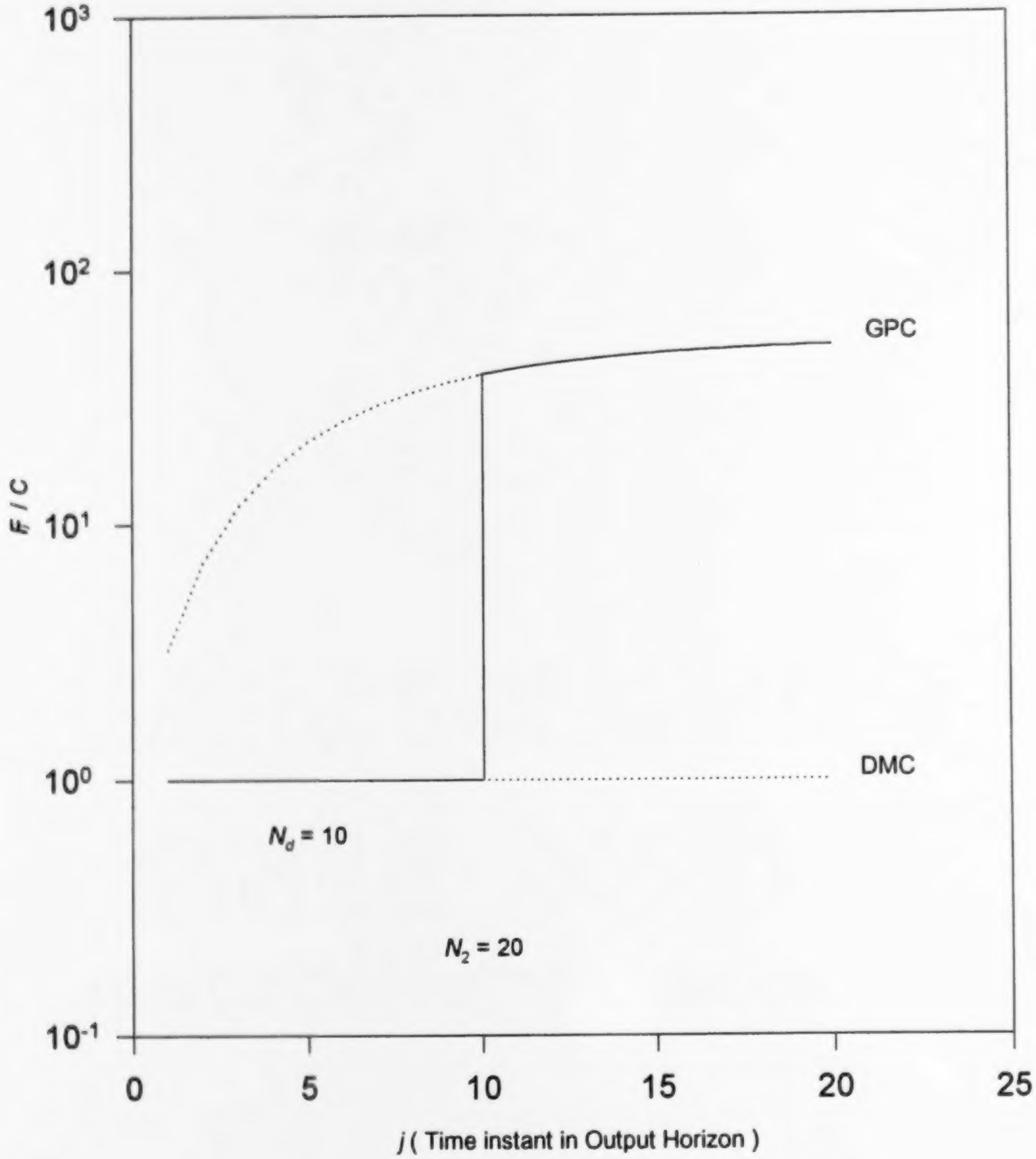


Figure 3.7 Uncertainty projection using disturbance horizon for DMC followed by GPC

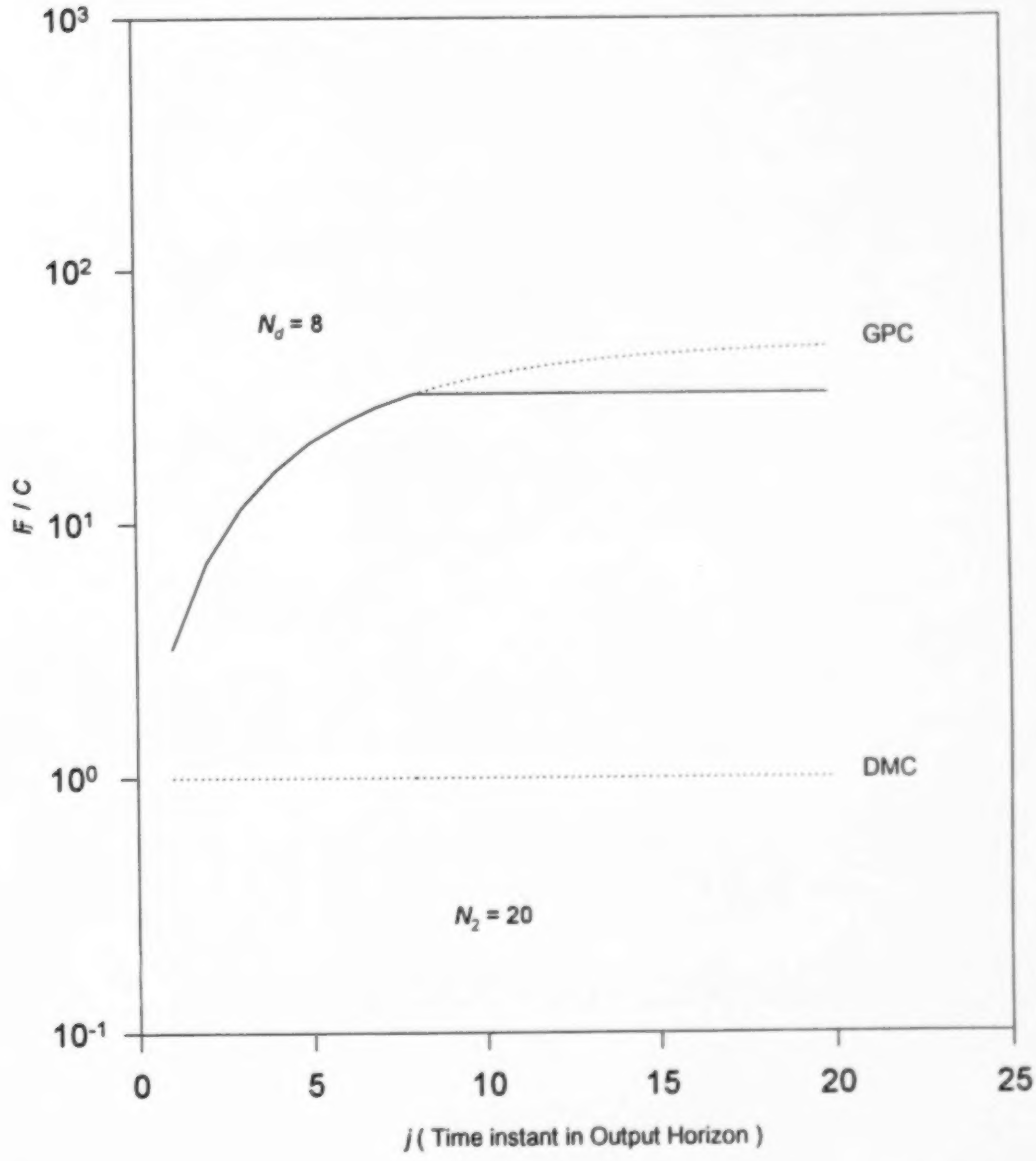


Figure 3.8 Uncertainty projection using the limited expansion of GPC noise model

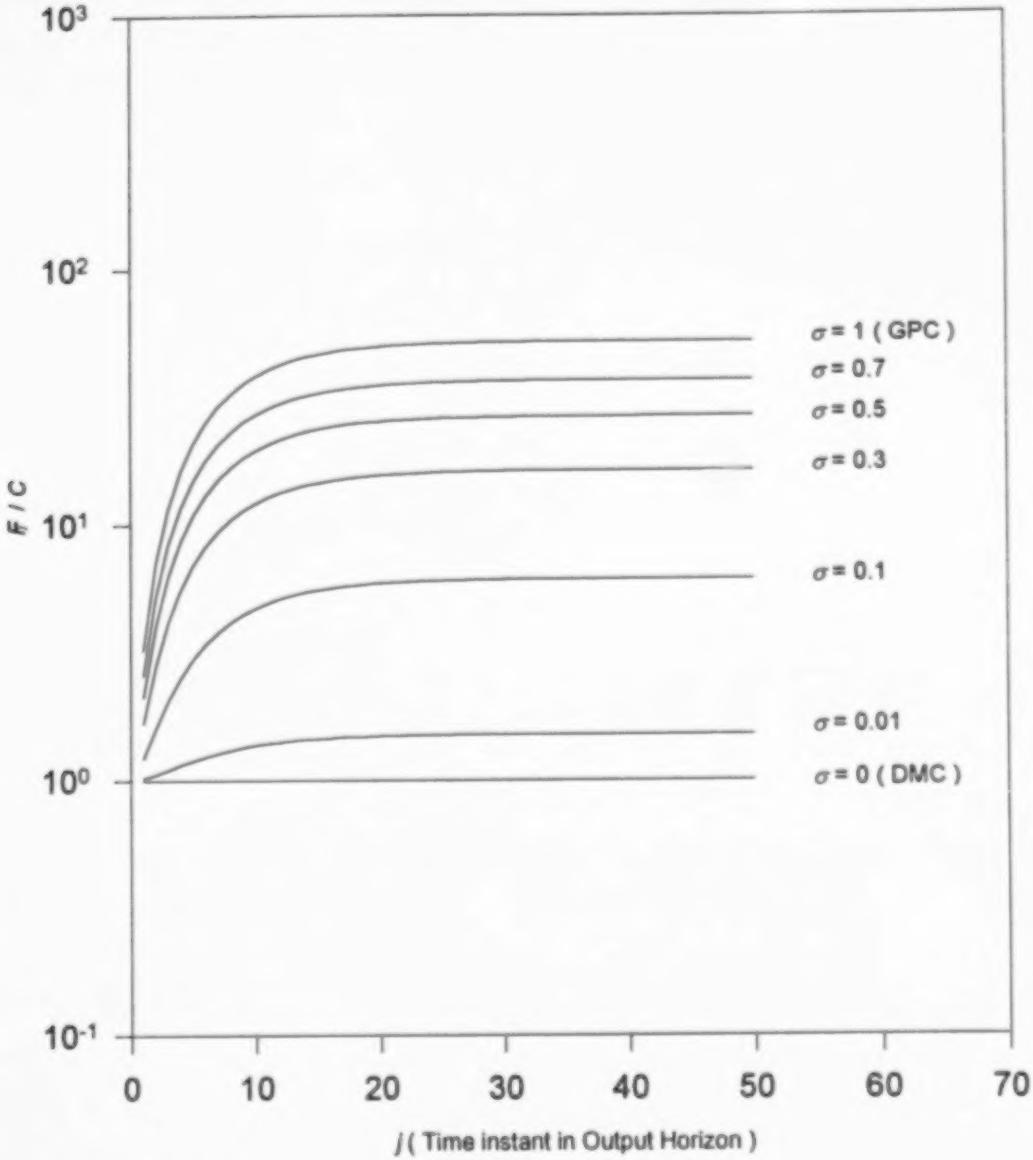


Figure 3.9 σ-Weighting between DMC and GPC noise models

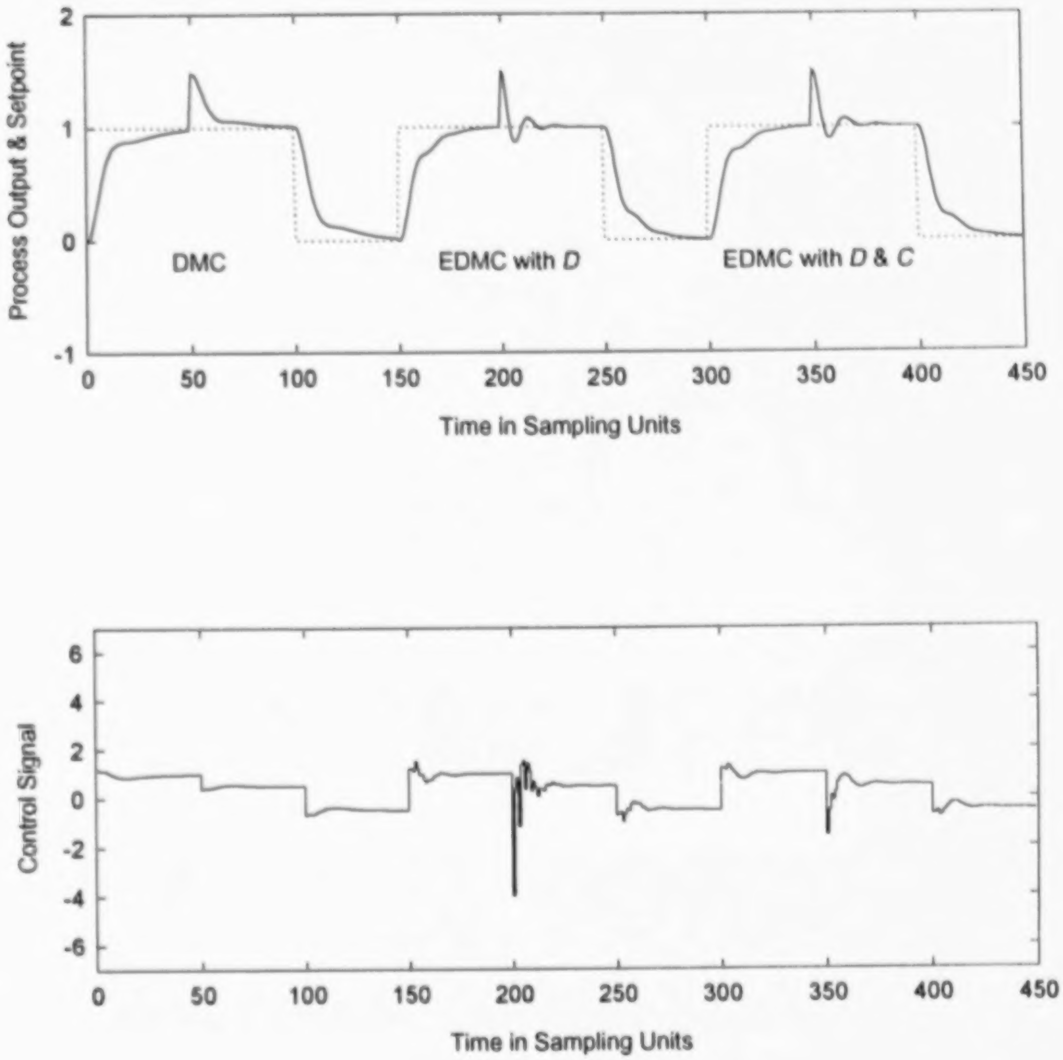


Figure 3.10 Disturbance rejection using enhanced DMC (EDMC)

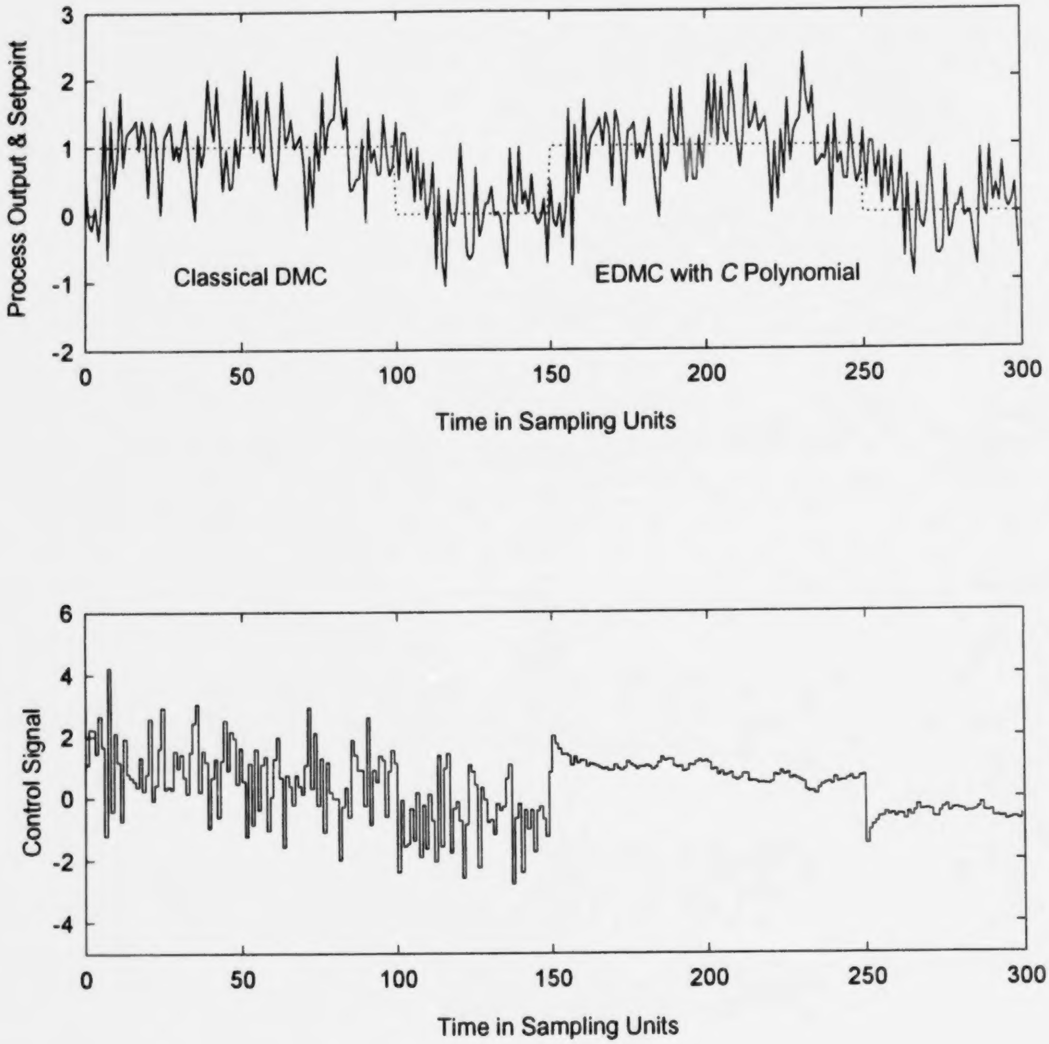


Figure 3.11 Effect of *C* polynomial on noise filtering in enhanced DMC (EDMC)

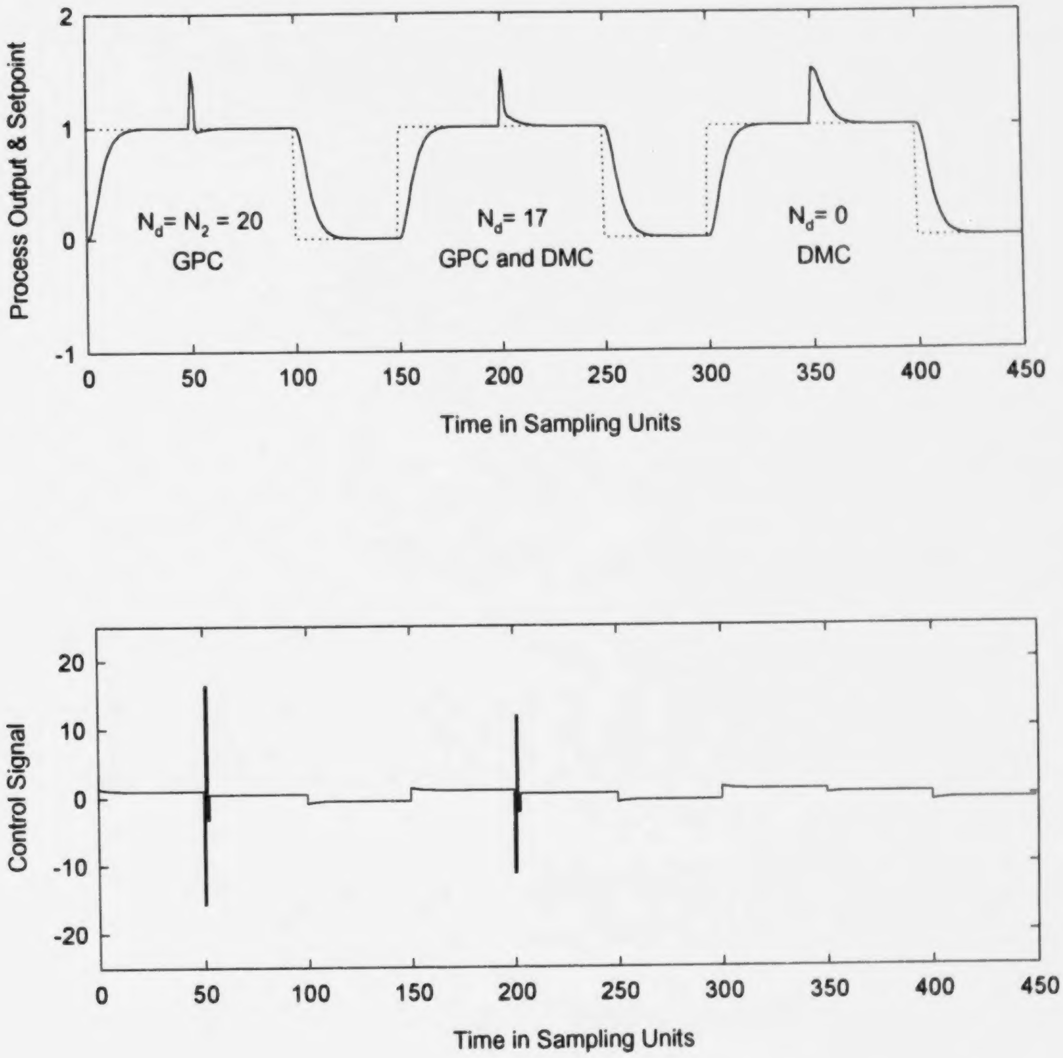


Figure 3.12 Incorporating two noise models (GPC & DMC) using disturbance horizon

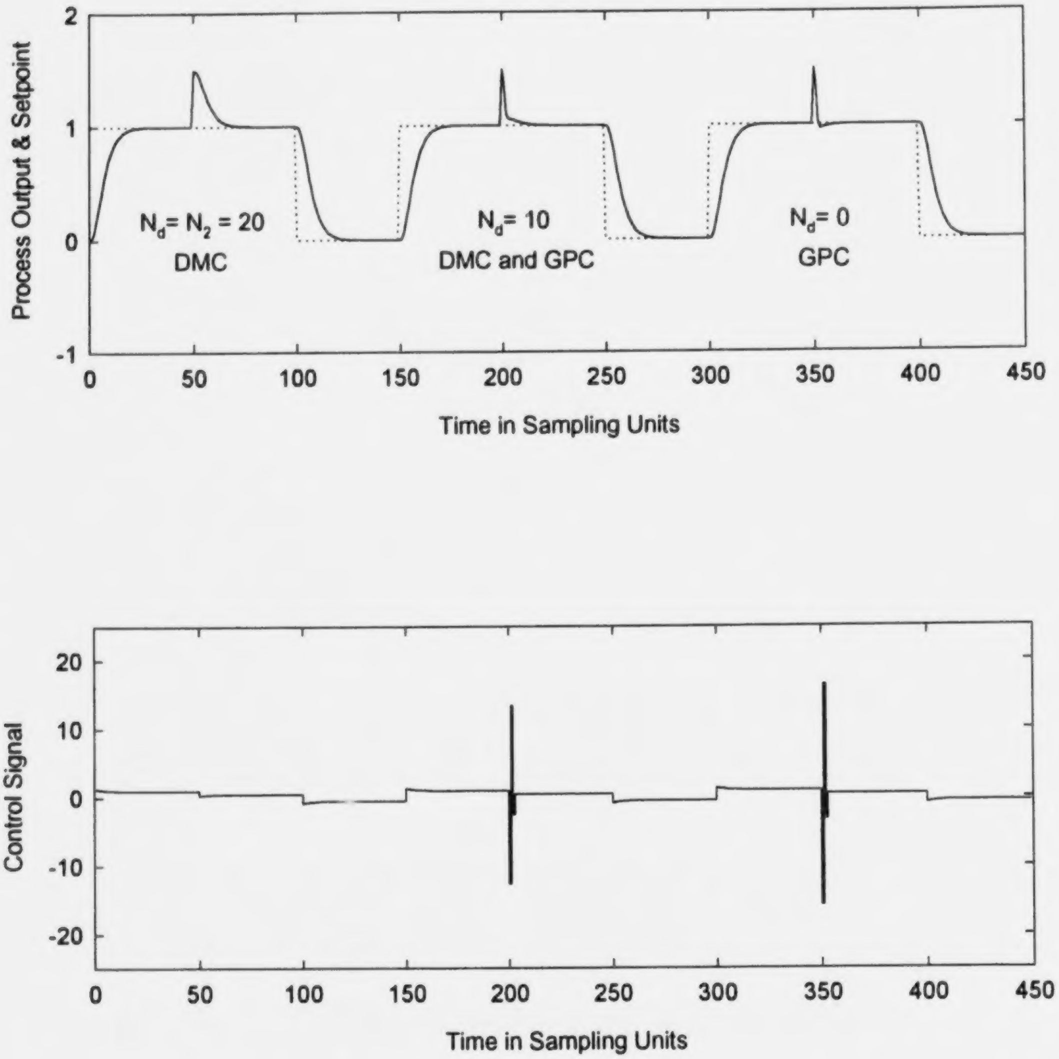
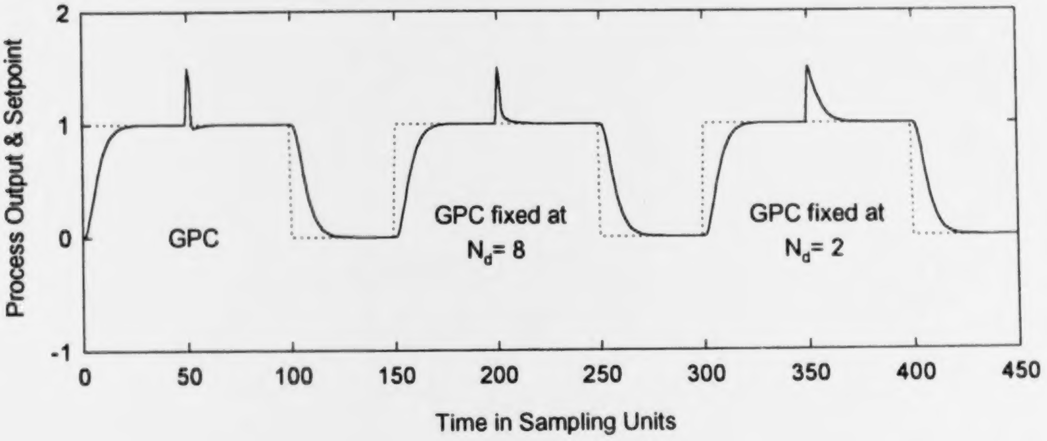


Figure 3.13 Incorporating two noise models (DMC & GPC) using disturbance horizon



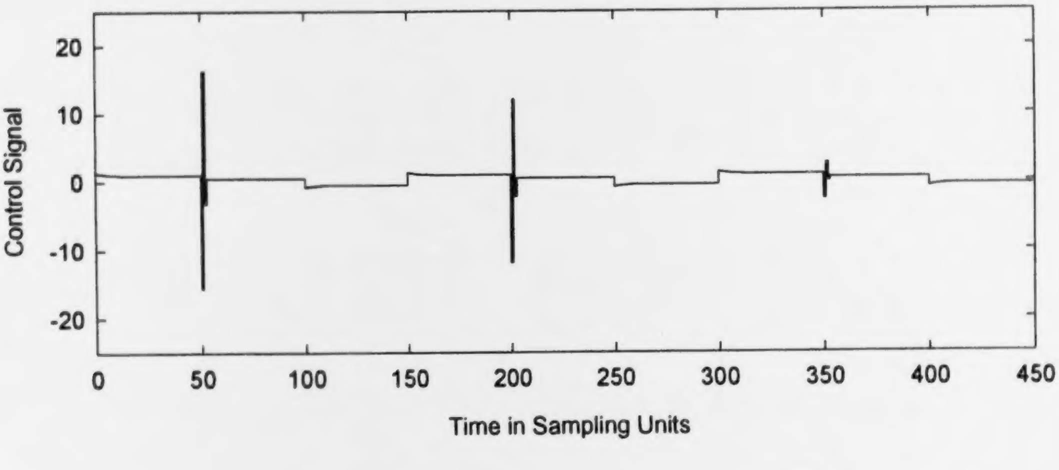


Figure 3.14 Limiting the GPC noise model expansion up to a disturbance horizon

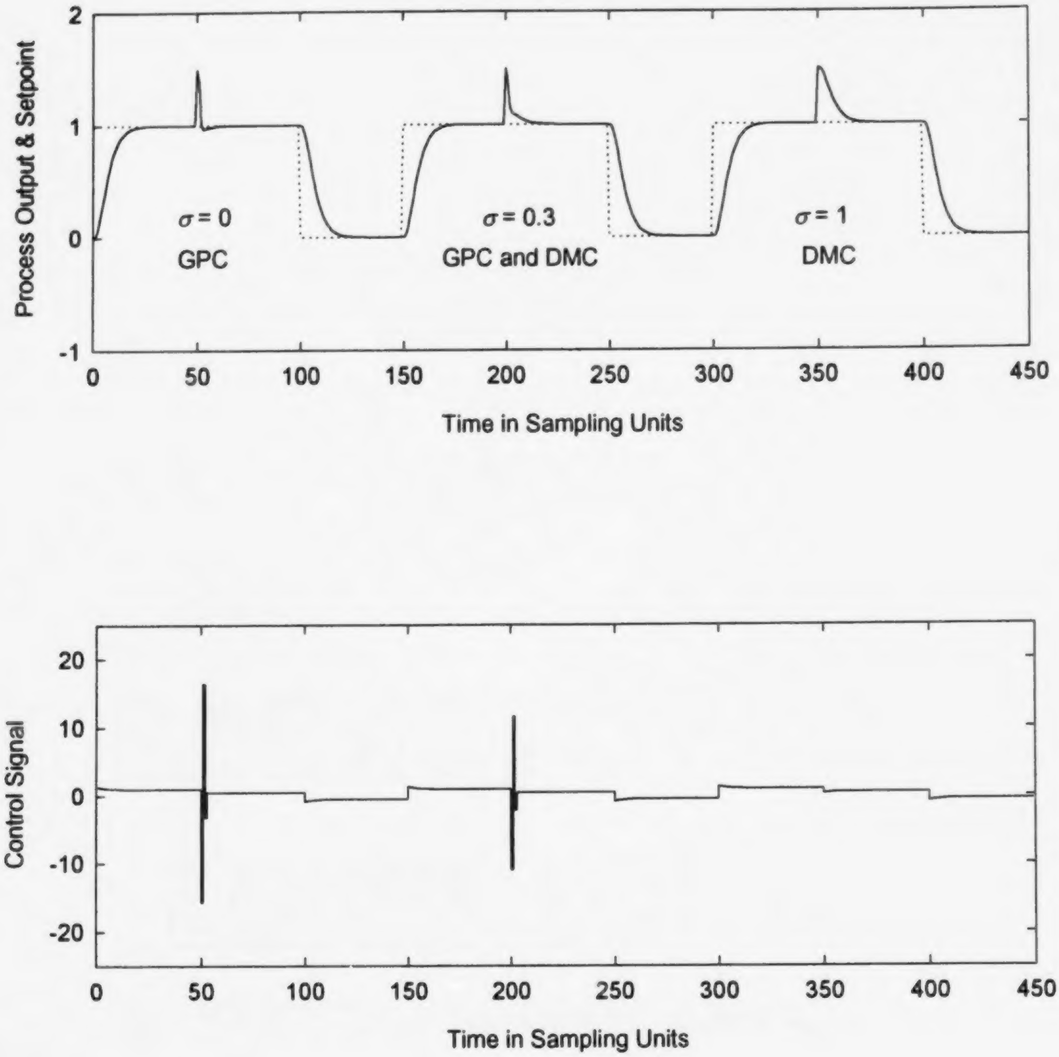


Figure 3.15 σ-Weighting to incorporate two noise models (GPC & DMC)

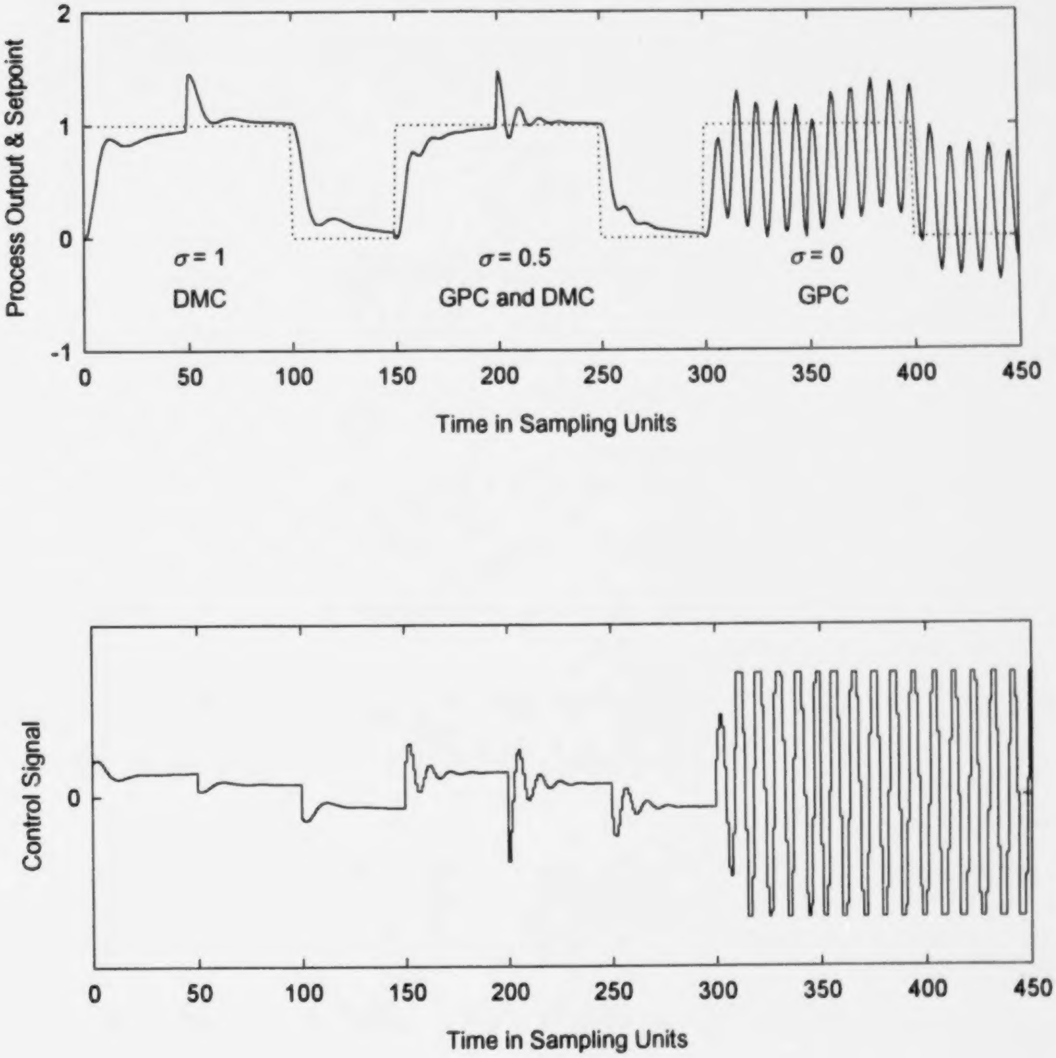


Figure 3.16 σ-Weighting to incorporate two noise models in the presence of MPM

Chapter 4

Reduced Diophantine Predictors for Long Range Predictive Control [1]

New reduced Diophantine predictors are developed and are proven to be mathematically equal to the conventional predictors used in the Long Range Predictive Controllers (LRPC's) such as GPC. The new formulations are much simpler, do not require filtered input and output values, and offer computational saving of up to 65% in adaptive applications. The proposed formulation also leads to new insight and interpretations of LRPC's.

4.1 Introduction

Long Range Predictive Controllers (LRPC's) are model-based optimal predictive control algorithms which use a predicted future output trajectory of the system to calculate a set of optimal control actions. The control implementation in LRPC's is called receding horizon control which means that only the first of the calculated control actions is applied and the procedure is repeated at each time instant so that the control is always calculated on the basis of the most recent system measurement.

A generalized measured output model for a transfer function based LRPC is:

$$y(t) = \frac{B}{A}u(t-1) + \frac{C}{D\tilde{A}\Delta^n}e(t) \tag{4.1}$$

---

[1] A version of this chapter was presented at the 1995 American Control Conference: Reduced Diophantine Predictors for Long Range Predictive Controllers, Saudagar, M.A., Fisher, D.G. and Shah, S.L., *Proc. American Control Conference*, pp. 3688, 1995.

where $y(t)$ = output at time, t ;$u(t)$ = input at t; $e(t)$ = zero-mean normally distributed independent white noise. A, B, C and D are polynomials in backward shift operator q^{-1}. For offset free response a multiple-integrator $1/\Delta^n$ ($\Delta=1-q^{-1}$) is included. The noise model may optionally contain the process model denominator A which is a characteristic of equation error noise model structures such as ARMAX and ARIMAX (Clarke, 1987). The notation $\breve{A}$ is used to indicate that the polynomial A shall only be included for equation error structures. More specifically:

$$\breve{A} = A \qquad \text{for EE structures} \tag{4.2}$$

$$\breve{A} = 1 \qquad \text{for OE structures} \tag{4.3}$$

The model employed in the popular Generalized Predictive Control (GPC) is a special case of (4.1). Setting $D = 1$, $\breve{A} = A$, $n = 1$ in the second (noise) term of (4.1) gives:

$$y(t) = \frac{B}{A}u(t-1) + \frac{C}{A\Delta}e(t) \tag{4.4}$$

Equation (4.1) may also be written as:

$$y(t) = \frac{B}{A}u(t-1) + x(t) \tag{4.5}$$

$$y(t) = y_m(t) + x(t) \tag{4.6}$$

where $y_m(t)$ is the model output given as follows:

$$y_m(t) = \frac{B}{A}u(t-1) \tag{4.7}$$

and $x(t)$, the residual, is a lumped term representing the difference at time, t of the measured output and the model output:

$$x(t) = y(t) - y_m(t) \tag{4.8}$$

Using (4.1) $x(t)$ is represented as a stochastic signal in terms of $e(t)$:

$$x(t) = \frac{C}{D\breve{A}\Delta^n}e(t) \tag{4.9}$$

In the case of zero noise and disturbance, $x(t)$ represents the MPM (Model Plant Mismatch) at time, t. In the general case $x(t)$ represents a lumped combination of noise, MPM and external disturbances.

The lumped Diophantine predictor (LDP), based on the model (4.1), for the j-step ahead output prediction was developed in Chapter 2 and is given as:

$$\boxed{y_p(t+j|t) = \tilde{G}_j \Delta^n u(t+j-1) + \overline{G}_j \Delta^n u^f(t-1) + F_j y^f(t) \qquad (4.10)\ \text{LDP}}$$

Where the following equations define the various filters and signals in the LDP:

$$\Delta^n u^f(t-1) = \frac{\Delta^n u(t-1)}{C\hat{A}} \quad \text{and} \quad y^f(t) = \frac{y(t)}{C} \tag{4.11}$$

$$\hat{A} = 1 \qquad \text{for EE structures} \tag{4.12}$$

$$\hat{A} = A \qquad \text{for OE structures} \tag{4.13}$$

$$\left[\frac{C}{D\breve{A}\Delta^n}\right] = E_j + q^{-j}\frac{F_j}{D\breve{A}\Delta^n} \tag{4.14}$$

$$\left[\frac{E_j DB}{C\hat{A}}\right] = \tilde{G}_j + q^{-j}\frac{\overline{G}_j}{C\hat{A}} \tag{4.15}$$

The first term on the RHS of (4.10) is called the *forced response* and all the other terms on the RHS constitute the *free response*. The LDP uses a lumped (in the sense it treats $\dfrac{C}{D\breve{A}\Delta^n}$ as a single entity) Diophantine equation (4.14)

Remark 4.1-1:

The LDP includes two sets of Diophantine expressions (equations (4.14) and (4.15))

Remark 4.1-2:

Filtered values of input and output ($\Delta^n u^f(t)$ and $y^f(t)$) must be stored for output predictions.

Remark 4.1-3:

The C polynomial is first included in the Diophantine expansion of equation (4.14), and later it is deconvolved in the second Diophantine expansion of equation (4.15) which suggest the inclusion of C in (4.14) is redundant as shown below.

4.2 The Reduced Diophantine Predictor (RDP)

If the numerator polynomial C is removed from the Diophantine expansion (4.14), the resulting Diophantine equation to be solved becomes:

$$\left[\frac{1}{DA\Delta^n}\right] = \tilde{V}_j + q^{-j}\frac{\bar{V}_j}{DA\Delta^n} \tag{4.16}$$

The predictor based on the above Diophantine equation is referred to as the Reduced Diophantine Predictor or RDP. The model (4.1) may be shifted by $+j$ and rearranged as:

$$y(t+j) = \left[\frac{1}{DA\Delta^n}\right]DB\Delta^n u(t+j-1) + \left[\frac{1}{DA\Delta^n}\right]C\hat{A}e(t+j) \tag{4.17}$$

Substituting equation (4.16) for $\left[\frac{1}{DA\Delta^n}\right]$ in (4.17) yields:

$$y(t+j) = \left[\tilde{V}_j + q^{-j}\frac{\bar{V}_j}{DA\Delta^n}\right]DB\Delta^n u(t+j-1) + \left[\tilde{V}_j + q^{-j}\frac{\bar{V}_j}{DA\Delta^n}\right]C\hat{A}e(t+j) \tag{4.18}$$

The degree of polynomial A= na, the degree of $\tilde{V}_j$ = j-1. Rearrangement of the above equation gives:

$$y(t+j) = \tilde{V}_j DB\Delta^n u(t+j-1) + \bar{V}_j\left[\frac{B}{A}u(t-1) + \frac{C}{D\breve{A}\Delta^n}e(t)\right] + \tilde{V}_j C\hat{A}e(t+j) \tag{4.19}$$

or substituting y(t) for the bracketed term using (4.1) yields:

$$y(t+j) = \tilde{V}_j DB\Delta^n u(t+j-1) + \bar{V}_j y(t) + \tilde{V}_j C\hat{A}e(t+j) \tag{4.20}$$

Now taking the expectation of both sides of the above equation and noting that the expected value of $e(t+j)$ is zero, the following equation for the j-step predictor is obtained:

$$y_p(t+j|t) = \tilde{V}_j DB\Delta^n u(t+j-1) + \bar{V}_j y(t) + \overline{\left[\tilde{V}_j C\hat{A}\right]}_j e(t) \qquad (4.21)$$

where $y_p(t+j|t)$ is the prediction of the output y for $t+j$ conditioned on the input/output data available at time t and $\overline{\left[\tilde{V}_j C\hat{A}\right]}_j$ is defined by the following expression:

$$\tilde{V}_j C\hat{A} = \left[\tilde{V}_j C\hat{A}\right]_j + q^{-j}\,\overline{\left[\tilde{V}_j C\hat{A}\right]}_j \qquad (4.22)$$

Equation (4.22) simply defines the separation of the polynomial $\tilde{V}_j C\hat{A}$ into terms up to order j and the remaining higher order terms. The following expression is then used to separate the forced and the free responses

$$\tilde{V}_j DB = \left[\tilde{V}_j DB\right]_j + q^{-j}\,\overline{\left[\tilde{V}_j DB\right]}_j \qquad (4.23)$$

The final form of the RDP is as follows:

$$\boxed{y_p(t+j|t) = \left[\tilde{V}_j DB\right]_j \Delta^n u(t+j-1) + \overline{\left[\tilde{V}_j DB\right]}_j \Delta^n u(t-1) + \bar{V}_j y(t) + \overline{\left[\tilde{V}_j C\hat{A}\right]}_j e(t) \quad (4.24) \quad (\mathrm{RDP})}$$

The term $e(t)$ is reconstructed as:

$$e(t) = \frac{D\breve{A}\Delta^n}{C}\left[y(t) - \frac{B}{A}u(t-1)\right] \qquad (4.25)$$

Remark 4.2-1:

The computational load of separating the past and present from future terms in equations (4.22) and (4.23) is trivial because it is equivalent to picking the first j elements of the corresponding arrays.

Remark 4.2-2:

Only one set of Diophantine expansion (given by equation (4.16)) is involved which results in lower computational load than required in the classical approach which requires two Diophantine equations for the LDP((4.14) and (4.15)).

Remark 4.2-3:

$\Delta u^f(t)$ and $y^f(t)$ (i.e. the filtered $\Delta u(t)$ and $y(t)$ in the LDP are not required. Instead a single extra variable $e(t)$ from (4.24) is used. This simplifies the theoretical analyses of the controller based on this predictor.

Remark 4.2.4:

The derivation and implementation are much simpler and straight forward.

Remark 4.2-5:

The C polynomial is not part of the Diophantine equation, and therefore C may be changed at any time with no extra control computation, e.g. to adjust the noise filtering on-line.

Remark 4.2-6:

The Reduced Diophantine Predictor (RDP) is mathematically identical to the classical LDP. This is shown by Theorem-1.

Theorem-1:

The RDP (4.24) is mathematically identical to the LDP (4.10)

Proof:

Equation (4.20) may be re-written as:

$$y(t+j) = \tilde{V}_j DB\Delta^n u(t+j-1) + \bar{V}_j y(t) + \left[\tilde{V}_j C\hat{A}\right]_j e(t+j) + \overline{\left[\tilde{V}_j C\hat{A}\right]}_j e(t) \tag{4.26}$$

Subtracting (4.26) from (4.20) yields:

$$y_p(t+j|t) - y(t+j) = -\left[\tilde{V}_j C\hat{A}\right]_j e(t+j) \tag{4.27}$$

Shifting t in (4.1) by $+j$ and using equations (4.14) gives:

$$y(t+j) = \frac{B}{A}u(t+j-1) + \frac{F_j}{D\breve{A}\Delta^n}e(t) + E_j e(t+j) \tag{4.28}$$

or substituting the above equation in (4.27) gives

$$y_p(t+j|t) = \frac{B}{A}u(t+j-1) + \frac{F_j}{D\breve{A}\Delta^n}e(t) + E_j e(t+j) - \left[\tilde{V}_j C\hat{A}\right]_j e(t+j) \tag{4.29}$$

Now consider the future component $\left[\tilde{V}_j C\hat{A}\right]_j$

$$E_j = \left[\{C\}_j \left\{\frac{1}{D\breve{A}\Delta^n}\right\}_j\right]_j = \left[\{C\}_j \{\hat{A}\}_j \left\{\frac{1}{DA\Delta^n}\right\}_j\right]_j = \left[\{C\}_j \{\hat{A}\}_j \tilde{V}_j\right]_j \qquad (4.30)$$

where

$$\left\{\frac{1}{DA\Delta^n}\right\}_j = \tilde{V}_j \qquad (4.31)$$

Therefore, based on the rules of shift algebra given in Appendix-A of this chapter

$$\left[\tilde{V}_j C\hat{A}\right]_j = E_j \qquad (4.32)$$

Substituting (32) in (28) gives:

$$y_p(t+j|t) = \frac{B}{A}u(t+j-1) + \frac{F_j}{D\breve{A}\Delta^n}e(t) \qquad (4.33)$$

from which equation (4.10) for the LDP follows.

◆◆◆

Remark 4.2-7:

One of the requirements for an offset-free process response is that the steady state prediction must converge to the current output value. The steady state prediction from, the Reduced Diophantine Predictor (RDP) is easily shown to be equal to the current output. This may be shown by using $\overline{V}_j = q^j\left[1 - D\breve{A}\Delta^n\tilde{V}_j\right]$in (4.24) to produce:

$$\begin{aligned} y_p(t+j|t) = \left[\tilde{V}_j DB\right]_j \Delta^n u(t+j-1) + \overline{\left[\tilde{V}_j DB\right]}_j \Delta^n u(t-1) \\ +q^j\left[1 - D\breve{A}\Delta^n\tilde{V}_j\right]y(t) + \overline{\left[\tilde{V}_j C\hat{A}\right]}_j e(t) \end{aligned} \qquad (4.34)$$

Note that there is at least one Δ in $e(t)$. Thus, at steady state $q^1 = 1$, or $\Delta = 0$ giving $y_p(t+j|t) = y(t)$.

Remark 4.2-8:

The proof that the servo response is independent of the *C* polynomial for the case of zero model-plant mismatch is very trivial with the new formulation. The *C* polynomial in the RDP appears only in the term with $e(t)$ and for the case of perfect modeling and noise-free servo response $e(t) = 0$, so the noise-free servo response is obviously independent of the *C* polynomial. The proof of this property requires a much longer derivation and argument when it is based on the conventional GPC predictor (McIntosh, 1988) which uses the LDP (4.10).

Remark 4.2-9:

The classical DMC predictor can be obtained as a special case of RDP by noting that for DMC $A = C = D = 1$, $B = H$ and $n = 1$.

The Diophantine equation (4.16) becomes:

$$\frac{1}{\Delta} = \tilde{V}_j + q^{-j}\frac{\bar{V}_j}{\Delta} = \frac{1-q^{-j}}{\Delta} + q^{-j}\frac{1}{\Delta} \tag{4.35}$$

which implies

$$\tilde{V}_j = \frac{1-q^{-j}}{\Delta} \qquad \text{and} \qquad \bar{V}_j = 1 \tag{4.36}$$

where

$$H = \sum_{i=1}^{\infty} h_i q^{-i+1} \qquad \text{where the } h_i \text{ are impulse - response coefficients} \tag{4.37}$$

Moreover

$$\begin{gathered} S = \frac{H}{\Delta} = \sum_{i=1}^{\infty} \frac{h_i}{\Delta} q^{-i+1} = \sum_{i=1}^{\infty} s_i q^{-i+1} \quad \text{where } s_i \text{ are step - response coefficients} \\ s_i = \sum_{k=1}^{i} h_k \qquad \text{and} \quad h_i = s_i - s_{i-1} \quad \text{with } s_0 = 0 \text{ (Seborg et al., 1989)} \end{gathered} \tag{4.38}$$

$$\tilde{V}_j DB = \frac{1-q^{-j}}{\Delta} H = (1-q^{-j})S \tag{4.39}$$

$$\left[1-q^{-j}\right]S = S - q^{-j}S = \tilde{S}_j + q^{-j}\left[\bar{S}_j - S\right] = \sum_{i=1}^{j} s_i q^{-i+1} + q^{-j} \sum_{i=j+1}^{\infty} \left(s_i - s_{i-j}\right) q^{-i+j+1} \tag{4.40}$$

where the following relation hs been used:

$$S = \tilde{S}_j + q^{-j}\bar{S}_j \qquad \tilde{S}_j = \sum_{i=1}^{j} s_i q^{-i+1} \qquad \bar{S}_j = \sum_{i=j+1}^{\infty} s_i q^{-i-j+1} \tag{4.41}$$

therefore

$$\left[\tilde{V}_j DB\right]_j = \sum_{i=1}^{j} s_i q^{-i+1} \qquad \text{and} \qquad \overline{\left[\tilde{V}_j DB\right]}_j = \sum_{i=j+1}^{\infty} \left(s_i - s_{i-j}\right) q^{-i+j+1} \tag{4.42}$$

also

$$\overline{\left[\tilde{V}_j C\hat{A}\right]}_j = \overline{\left[\tilde{V}_j\right]}_j = 0 \tag{4.43}$$

Substituting the above in the RDP equation (4.24) gives:

$$y_p(t+j|t) = \left[\sum_{i=1}^{j} s_i q^{-i+1}\right]\Delta u(t+j-1) + \left[\sum_{i=j+1}^{\infty}\left(s_i - s_{i-j}\right)q^{-i+j+1}\right]\Delta u(t-1) + y(t) \tag{4.44}$$

or

$$y_p(t+j|t) = \left[\sum_{i=1}^{j} s_i \Delta u(t+j-i)\right] + \left[\sum_{i=j+1}^{\infty}\left(s_i - s_{i-j}\right)\Delta u(t-i+j)\right] + y(t) \tag{4.45}$$

Truncating the second sum at T, the settling time, gives the following DMC predictor:

$$y_p(t+j|t) = \left[\sum_{i=1}^{j} s_i \Delta u(t+j-i)\right] + \left[\sum_{i=j+1}^{T}\left(s_i - s_{i-j}\right)\Delta u(t-i+j)\right] + y(t) \tag{4.46}$$

Using (4.5), DMC predictor in terms of residual $x(t)$ is obtained as:

$$y_p(t+j|t) = \left[\sum_{i=1}^{j} s_i \Delta u(t+j-i)\right] + \left[\sum_{i=j+1}^{\infty} s_i \Delta u(t+j-i)\right] + x(t) \tag{4.47}$$

which after truncation becomes:

$$\boxed{y_p(t+j|t) = \left[\sum_{i=1}^{j} s_i \Delta u(t+j-i)\right] + \left[\sum_{i=j+1}^{T} s_i \Delta u(t+j-i)\right] + s_T u(t+j-T-1) + x(t)} \tag{4.48}$$

Remark 4.2-10:

The RDP equation (4.24) gives, with $\breve{A} = A$, $D = 1$, and $n = 1$, the following predictor for GPC with T-filter (C polynomial) :

$$y_p(t+j|t) = \left[\tilde{V}_j B\right]_j \Delta u(t+j-1) + \overline{\left[\tilde{V}_j B\right]}_j \Delta u(t-1) + \bar{V}_j y(t) + \overline{\left[\tilde{V}_j C\right]}_j e(t) \tag{4.49}$$

Note that the C polynomial appears only in the last term which vanishes if the degree of C is zero. This gives a predictor for GPC with T-filter as a sum of the predictor for GPC without T-filter and an additive term as:

$$y_p(t+j|t)_{\text{GPC with T-filter}} = y_p(t+j|t)_{\text{GPC without T-filter}} + \overline{\left[\tilde{V}_j C\right]}_j e(t) \tag{4.50}$$

The advantage of the above additive expression is that the T-filter contribution to the output prediction is given separately which might be useful in determining its robustness properties and other analyses. This decoupling of T-filter contribution is not available in the classical GPC predictor (LDP) as shown below.

The original predictor for GPC with T-filter (C polynomial) is obtained using LDP for UMPC model with $\breve{A} = A$, $D = 1$, and $n = 1$, and is given as:

$$y_p(t+j|t)_{\text{GPC with T-filter}} = \tilde{G}_j \Delta u(t+j-1) + \overline{G}_j \frac{\Delta u(t-1)}{C} + F_j \frac{y(t)}{C} \tag{4.51}$$

The GPC predictor without T-filter is:

$$y_p(t+j|t)_{\text{GPC without T-filter}} = \tilde{G}_j^{'} \Delta u(t+j-1) + \overline{G}_j^{'} \Delta u(t-1) + F_j^{'} y(t) \tag{4.52}$$

Note that in case of LDP, $y_p(t+j|t)_{\text{GPC with T-filter}}$ can not be given in terms of $y_p(t+j|t)_{\text{GPC without T-filter}}$.

The RDP is graphically presented in the block diagram given in Figure 4.1. In this figure the predictor does not give separate free responses due to the manipulated variable and uncertainty. In Figure 4.2 the free response for the RDP is shown to be a sum of the free responses due to the past manipulated variable and uncertainty.

4.3 The Reduced Diophantine Overparameterized Predictor (RDOP):

Solution of the Diophantine equations, in the LRPC algorithms such as GPC, requires huge computational and storage resources (Kramer, 1991). One modification to reduce the computational resources is overparameterization of the output predictor developed in the following.

The separation of future and past components using the Diophantine equation (4.16) for $t+j$ is not unique. A Diophantine equation in terms of $N \geq j$ may be used. For example, the predictor can be simplified by using the following single Diophantine identity for $N \geq N2$ to replace the Diophantine equations for all individual *j*'s:

$$\frac{1}{DA\Delta^n} = \tilde{V}_N + q^{-N}\frac{\bar{V}_N}{DA\Delta^n} \tag{4.53}$$

The predictor based on the Diophantine equation (4.53) is overparameterized and therefore referred to as Reduced Diophantine Overparameterized Predictor (RDOP).

Substituting equation (4.53) for $\left[\dfrac{1}{DA\Delta^n}\right]$ in (4.17) gives:

$$y(t+j) = \left[\tilde{V}_N + q^{-N}\frac{\bar{V}_N}{DA\Delta^n}\right]DB\Delta^n u(t+j-1) + \left[\tilde{V}_N + q^{-N}\frac{\bar{V}_N}{DA\Delta^n}\right]C\hat{A}e(t+j) \tag{4.54}$$

The degree of $\tilde{V}_N = N\text{-}1$ so the above equation can be rearranged to

$$\begin{aligned} y(t+j) = \tilde{V}_N DB\Delta^n u(t+j-1) + \tilde{V}_N C\hat{A}e(t+j) \\ + \bar{V}_N\left[\frac{B}{A}u(t-N+j-1) + \frac{C}{D\breve{A}\Delta^n}e(t-N+j)\right] \end{aligned} \tag{4.55}$$

The terms in square brackets is $y(t\text{-}N\text{+}j)$ so

$$y(t+j) = \tilde{V}_N DB\Delta^n u(t+j-1) + \bar{V}_N y(t-N+j) + \tilde{V}_N C\hat{A}e(t+j) \tag{4.56}$$

After taking the expectations:

$$y_p(t+j|t) = \tilde{V}_N DB\Delta^n u(t+j\text{-}1) + \bar{V}_N y(t-N+j) + \overline{\left[\tilde{V}_N C\hat{A}\right]}_j e(t) \tag{4.57}$$

where $\overline{\left[\tilde{V}_N C\hat{A}\right]}_j$ is defined by the following expression:

$$\tilde{V}_N C\hat{A} = \left[\tilde{V}_N C\hat{A}\right]_j + q^{-j}\overline{\left[\tilde{V}_N C\hat{A}\right]}_j \tag{4.58}$$

where $\left[\tilde{V}_N C\hat{A}\right]_j$ represents the first j terms of $\tilde{V}_N C\hat{A}$ and $\overline{\left[\tilde{V}_N C\hat{A}\right]}_j$ consists of the remaining terms.

The forced and free responses are separated using the following expression

$$\tilde{V}_N DB = \left[\tilde{V}_N DB\right]_j + q^{-j}\overline{\left[\tilde{V}_N DB\right]}_j \tag{4.59}$$

where $\left[\tilde{V}_N DB\right]_j$ represents the first j terms of $\tilde{V}_N DB$ and $\overline{\left[\tilde{V}_N DB\right]}_j$ consists of the remaining terms.

The final form of the RDOP is then:

$$\boxed{\begin{aligned} y_p(t+j|t) = \left[\tilde{V}_N DB\right]_j \Delta^n u(t+j-1) + \overline{\left[\tilde{V}_N DB\right]}_j \Delta^n u(t-1) & \\ + \bar{V}_N y(t-N+j) + \overline{\left[\tilde{V}_N C\hat{A}\right]}_j e(t) & \qquad (4.60) \\ & \qquad \text{(RDOP)} \end{aligned}}$$

Remark 4.3-1:

The computational load associated with the separation in the equations (4.58) and (4.59) is trivial because it is equivalent to picking the first *j* elements of the corresponding arrays.

Remark 4.3-2:

Only a single Diophantine expansion (given by equation (4.53)) is used for the entire prediction horizon as compared to the LDP and the RDP which require two sets and one set of Diophantine expansion respectively where there are *N2* equations in each set. This results in substantial saving in computer storage and computational load. Reductions of more than 65% have been observed in simulation examples.. Specifically the large computational saving is due to following the two factors:

i) The entire Diophantine expansion is accomplished by a single call to a function such as *deconv* in Matlab.
ii) The N_2-N_1+1 convolution operations $E_j B$ are reduced to two convolution operations $\tilde{V}_N B$ and $\tilde{V}_N C$

The 65% computational saving is obtained under the following conditions:

- The computer programs have been written in Matlab command language.
- The controller is adaptive, i.e. A and B are estimated at each control interval
- The percentage computational saving is based on only the control law calculations block of the Matlab programs.
- The controller is designed for the Reduced Diophantine Overparameterized predictor.

Remark 4.3-3:

Since the reduction in the computational load is exclusively achieved through the reformulation of the predictor, it is still valid for the controller with input/output constraints.

Remark 4.3-4:

The RDOP is mathematically identical to the classical LDP and the RDP. This is shown by Theorem-2.

Theorem-2:

The RDOP (4.60) is mathematically identical to the LDP (4.10)

Proof:

Equation (4.56) may be re-written as:

$$\begin{aligned} y(t+j) = \tilde{V}_N DB\Delta^n u(t+j-1) + \bar{V}_N y(t-N+j) \\ + \left[\tilde{V}_N C\hat{A}\right]_j e(t+j) + \overline{\left[\tilde{V}_N C\hat{A}\right]}_j e(t) \end{aligned} \tag{4.61}$$

Subtracting (4.61) from (4.57) gives:

$$y_p(t+j|t) - y(t+j) = -\left[\tilde{V}_N C\hat{A}\right]_j e(t+j) \tag{4.62}$$

But $y(t+j)$ is given in (4.28) as:

$$y(t+j) = \frac{B}{A} u(t+j-1) + \frac{F_j}{A\Delta} e(t) + E_j\, e(t+j) \tag{4.28}$$

Substituting the above equation in (4.61) gives

$$y_p(t+j|t) = \frac{B}{A} u(t+j-1) + \frac{F_j}{D\breve{A}\Delta^n} e(t) + E_j\, e(t+j) - \left[\tilde{V}_N C\hat{A}\right]_j e(t+j) \tag{4.63}$$

Now using shift algebra (reviewed in Appendix-A) and equation (4.32) it may easily be shown that:

$$\left[\tilde{V}_N C\hat{A}\right]_j = \left[\tilde{V}_j C\hat{A}\right]_j = E_j \tag{4.64}$$

therefore substituting (4.64) in (4.63) gives:

$$y_p(t+j|t) = \frac{B}{A} u(t+j-1) + \frac{F_j}{D\breve{A}\Delta^n} e(t) \tag{4.65}$$

from which equation (4.10) for LDP follows. ♦♦♦

Remark 4.3-5:

The proof of Theorem-2 is applicable for any $N \geq j$. Therefore as a special case it also serves as a proof to theorem-1 for $N = j$.

Remark 4.3-6:

The above derivation shows that the RDOP is independent of N. In fact $N > j$ has been primarily used to obtain a predictor involving a single Diophantine equation for the entire output horizon. This objective is achieved by setting $N = N2$. Any value of $N > N2$ while permissible, would result in somewhat more computational load. Hence $N = N2$ is recommended.

Remark 4.3-7:

Since the RDOP is equivalent to the RDP and LDP, it is also an optimal predictor. Therefore the saving in the computational load is obtained without sacrificing optimality.

4.4 The Control Law

The purpose of a long-range predictive controller is to find the current (at time = t) control action by considering the effect of the current plus (N_u-1) future control moves on the output over a future horizon bounded by times $t+N_1$ and $t+N_2$. Time $t+N_1$ is greater than or equal to the earliest time in the future at which the output is affected by the control move at time t, while N_2 is arbitrary but should be greater than or equal to N_1.

To date most of the popular long-range predictive controllers use the following type of cost function for the optimal control calculation (Clarke, 1987a,b):

$$J = \sum_{j=N_1}^{N_2} \gamma(j)\left[y_p(t+j|t) - w(t+j)\right]^2 + \sum_{j=1}^{N_u} \lambda(j)\left[\Delta^n u(t+j-1)\right]^2 \tag{4.66}$$

where

$\gamma(.)$ = the weighting on tracking error.

$y_p(.)$ = the predicted output values.

$w(.)$= the desired output values or the set-points.

$N_1 =$ the initial output horizon ≥ the earliest future output that is affected by the control move at time t.

$N_2 =$ the final output horizon, the farthest future output that is included in the cost function J.

$\lambda(.) =$ the control weighting sequence.

$N_u =$ the control horizon, the number of future control moves included in the cost function J.

The $y_p(t+j|t)$ in the above cost function is usually given by the LDP, RDP or the RDOP developed above and are repeated here for reference:

$$y_p(t+j|t) = \tilde{G}_j \Delta^n u(t+j-1) + \overline{G}_j \Delta^n u^f(t-1) + F_j y^f(t) \tag{4.10)}$$

LDP

$$y_p(t+j|t) = \left[\tilde{V}_j DB\right]_j \Delta^n u(t+j-1) + \overline{\left[\tilde{V}_j DB\right]}_j \Delta^n u(t-1) + \overline{V}_j y(t) + \overline{\left[\tilde{V}_j C\hat{A}\right]}_j e(t) \tag{4.24}$$

RDP

$$\begin{aligned} y_p(t+j|t) = \left[\tilde{V}_N DB\right]_j \Delta^n u(t+j-1) + \overline{\left[\tilde{V}_N DB\right]}_j \Delta^n u(t-1) \\ + \overline{V}_N y(t-N+j) + \overline{\left[\tilde{V}_N C\hat{A}\right]}_j e(t) \end{aligned} \tag{4.60}$$

RDOP

The prediction consists of two parts. The first term on the RHS of the predictor is the response due to the future input moves. This first part is known as *forced response*. The remaining terms on the RHS of the predictors are called the *free-response*. It is the response due to the past inputs $u(.)$ and disturbances up to time t.

The forced response is given as follow:

$$\begin{aligned} &\text{Forced Response for RDP } = \left[\tilde{V}_j DB\right]_j \Delta^n u(t+j-1) \\ &\text{Forced Response for RDOP } = \left[\tilde{V}_N DB\right]_j \Delta^n u(t+j-1) \\ &\text{Forced Response for LDP } = \tilde{G}_j \Delta^n u(t+j-1) \end{aligned} \tag{4.67}$$

where

$$\left[\tilde{V}_j DB\right]_j = \left[\tilde{V}_n DB\right]_j = \tilde{G}_j = g_0 + g_1 q^{-1} + \ldots + g_{j-1} q^{-j+1} \tag{4.68}$$

The free response is given as follow:

LDP free response:

$$f(t+j) = \overline{G}_j \Delta^n u^f(t-1) + F_j y^f(t) \tag{4.69}$$

RDP free response:

$$f(t+j) = \overline{\left[\tilde{V}_j DB\right]}_j \Delta^n u(t-1) + \overline{V}_j y(t) + \overline{\left[\tilde{V}_j C\hat{A}\right]}_j e(t) \tag{4.70}$$

RDOP free response:

$$f(t+j) = \overline{\left[\tilde{V}_N DB\right]}_j \Delta^n u(t-1) + \overline{V}_N y(t-N+j) + \overline{\left[\tilde{V}_N C\hat{A}\right]}_j e(t) \tag{4.71}$$

The control law is given as (McIntosh, 1988):

$$\mathbf{u} = \left[\mathbf{G}^{\mathrm{T}} \Gamma \mathbf{G} + \Lambda\right]^{-1} \mathbf{G}^{\mathrm{T}} \Gamma (\mathbf{w} - \mathbf{f}) \tag{4.72}$$

where

$$\mathbf{u} = \left[\Delta u(t) \quad \Delta u(t+1) \quad \cdots \quad \Delta u(t+Nu-1)\right]^{\mathrm{T}} \tag{4.73}$$

$$\mathbf{w} = \left[w(t+N_1) \quad w(t+N_1+1) \quad \cdots \quad w(t+N_2)\right]^{\mathrm{T}} \tag{4.74}$$

$$\mathbf{f} = \left[f(t+N_1) \quad f(t+N_1+1) \quad \cdots \quad f(t+N_2)\right]^{\mathrm{T}} \tag{4.75}$$

$$\Gamma = \mathrm{diag}\left[\gamma(N_1) \quad \gamma(N_1+1) \quad \cdots \quad \gamma(N_2)\right] \tag{4.76}$$

$$\Lambda = \mathrm{diag}\left[\lambda(1) \quad \lambda(2) \quad \cdots \quad \lambda(N_u)\right] \tag{4.77}$$

$$\mathbf{G} = \begin{bmatrix} g_{N_1-1} & g_{N_1-2} & \cdots & g_0 & 0 & \cdots & 0 \\ g_{N_1} & g_{N_1-1} & \cdots & g_1 & g_0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & & \ddots & g_0 \\ \vdots & \vdots & \ddots & \ddots & & & g_1 \\ \vdots & \vdots & & \ddots & & & \vdots \\ g_{N_2-1} & g_{N_2-2} & \cdots & \cdots & \cdots & \cdots & g_{N_2-N_u} \end{bmatrix} \tag{4.78}$$

4.5 Conclusions:

- A reduced Diophantine predictor (RDP) for Long Range Predictive Control (LRPC) is developed which involves only one, much simpler, set of Diophantine equations rather than the two sets used in the conventional GPC formulation (Clarke, 1987).
- The reduced Diophantine predictor (RDP) is not formulated in terms of filtered input and output values. Only the single calculated variable, $e(.)$ is required.
- The reduced Diophantine predictor (RDP) is proven to be mathematically equivalent to the conventional (GPC) lumped Diophantine predictor (LDP) and is therefore optimal.
- LRPC based on the reduced Diophantine predictor (RDP) has been found to be much simpler and computationally less expensive than the conventional lumped Diophantine predictor (LDP) based formulation.
- A reduced Diophantine overparameterized predictor (RDOP) is also derived and proven to be optimal and mathematically equivalent to the reduced Diophantine predictor (RDP) and the conventional lumped Diophantine predictor (LDP).
- The reduced Diophantine overparameterized predictor (RDOP) offers substantial computation savings and is therefore recommended over the LDP and RDP for LRPC's such as GPC and UPC.

References:

Clarke, D.W., Mohtadi, C. and Tuffs, P.S., " Generalized Predictive Control - Part I the Basic Algorithm ", Automatica, Vol. 23, No. 2, pp.137-148, 1987a.

Clarke, D.W., Mohtadi, C. and Tuffs, P.S., " Generalized Predictive Control - Part II Extensions and Interpretations ", Automatica, Vol. 23, No. 2, pp. 149-160, 1987b.

Kramer, K. and Unbehauen, H., "Predictive Adaptive Control - Comparison of Main Algorithms", Proc. of the First European Control Conference, pp. 327-332, 1991.

McIntosh, A.R., "Performance and Tuning of Adaptive Generalized Predictive Control ", M.Sc. thesis, University of Alberta, 1988.

Seborg, D.E., Edgar, T.F. and Mellichamp, D.A., "Process Dynamics and Control", pp. 652, John Wiley & Sons, 1989.

Appendix 4-A

Polynomial Shift Algebra

In the formulation and analysis of discrete-time long range predictors and controllers it is always necessary to divide various system polynomials and/or ratios of polynomials in q^{-1} into future and present-plus-past contributions whenever such polynomials (or the ratios thereof) are used with the signals which explicitly contain future values.

In the following development some basic facts and relationships for polynomial bifurcation are established. The expressions are developed for monic polynomials but this involves no loss of generality since any polynomial can easily be converted to a monic polynomial by simple scalar division.

Let:

$$\begin{aligned} A(q^{-1}) &= 1 + a_1 q^{-1} + a_2 q^{-2} + \cdots + a_{na} q^{-na} & \deg\{A(q^{-1})\} &= na \\ B(q^{-1}) &= 1 + b_1 q^{-1} + b_2 q^{-2} + \cdots + b_{nb} q^{-nb} & \deg\{B(q^{-1})\} &= nb \\ C(q^{-1}) &= 1 + c_1 q^{-1} + c_2 q^{-2} + \cdots + c_{nc} q^{-nc} & \deg\{C(q^{-1})\} &= nc \\ D(q^{-1}) &= 1 + d_1 q^{-1} + d_2 q^{-2} + \cdots + d_{nd} q^{-nd} & \deg\{D(q^{-1})\} &= nd \end{aligned}$$

Now assuming that the signal of interest is *j*-steps ahead in future (e.g. $y(t+j)$), then:

$$A(q^{-1}) = \left[A(q^{-1})\right]_j + q^{-j}\overline{\left[A(q^{-1})\right]}_j$$

Future Part   Present / Past

$$\deg[A(q^{-1})]_j = j - 1 \qquad \deg\overline{[A(q^{-1})]}_j = na - j$$

Similar expressions may be written for $B(q^{-1})$, $C(q^{-1})$ and $D(q^{-1})$ polynomials. In the following development the argument q^{-1} of the polynomials is omitted for clarity.

$$\left[AB\right]_j = \left[A_j B_j\right]_j = \left[A B_j\right]_j = \left[A_j B\right]_j$$

The above relation holds because any term in either A or B that is farther in the past than q^{-j+1} simply adds the corresponding terms of convolution in q^{-j} or farther in the past and ultimately these terms are truncated.

It also follows that:

$$[A\,B\,C\,D]_j = [A'\,B'\,C'\,D']_j$$

$$\text{where} \quad A' = A \text{ or } A_j \quad B' = B \text{ or } B_j \quad C' = C \text{ or } C_j \quad D' = D \text{ or } D_j$$

In other words either the polynomials themselves or their j-step future parts may be used inside the RHS square brackets. Similar results hold for inverse filters as given in the following expression:

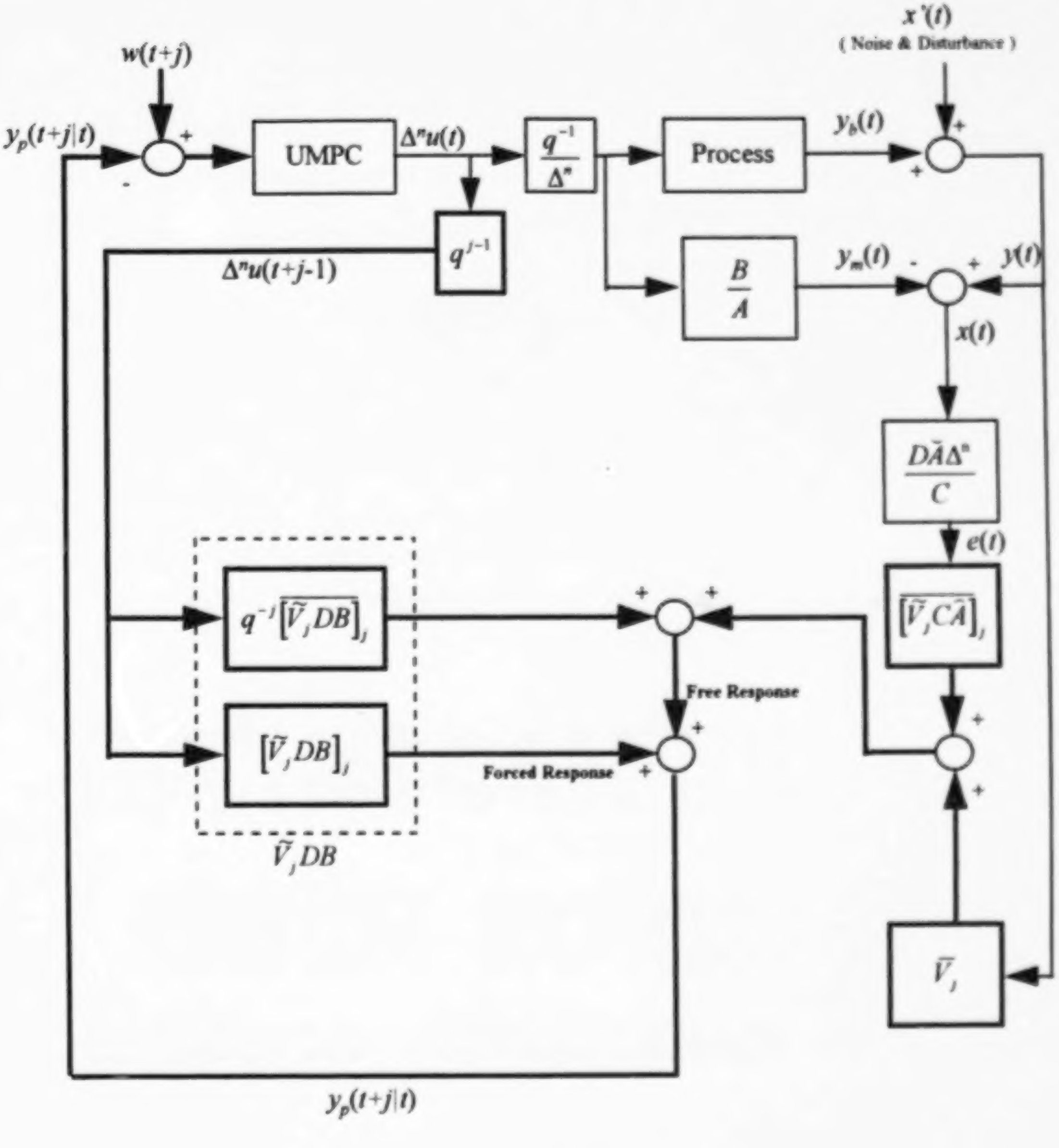
$$\left[\frac{1}{DA}\right]_j = \left[\left\{\frac{1}{D}\right\}\left\{\frac{1}{A}\right\}\right]_j = \left[\left\{\frac{1}{D}\right\}_j\left\{\frac{1}{A}\right\}\right]_j = \left[\left\{\frac{1}{D}\right\}\left\{\frac{1}{A}\right\}_j\right]_j = \left[\left\{\frac{1}{D}\right\}_j\left\{\frac{1}{A}\right\}_j\right]_j$$

For rational filters (i.e. the ratio of polynomials) the following relations hold:

$$\left[\frac{C}{A}\right] = \left[\frac{C}{A}\right]_j + q^{-j}\overline{\left[\frac{C}{A}\right]}_j$$

$$\left[\frac{C}{A}\right]_j = \left[C_j\left\{\frac{1}{A}\right\}_j\right]_j$$

$$\overline{\left[\frac{C}{A}\right]}_j = q^j\left(\left[\frac{C}{A}\right] - \left[\frac{C}{A}\right]_j\right)$$

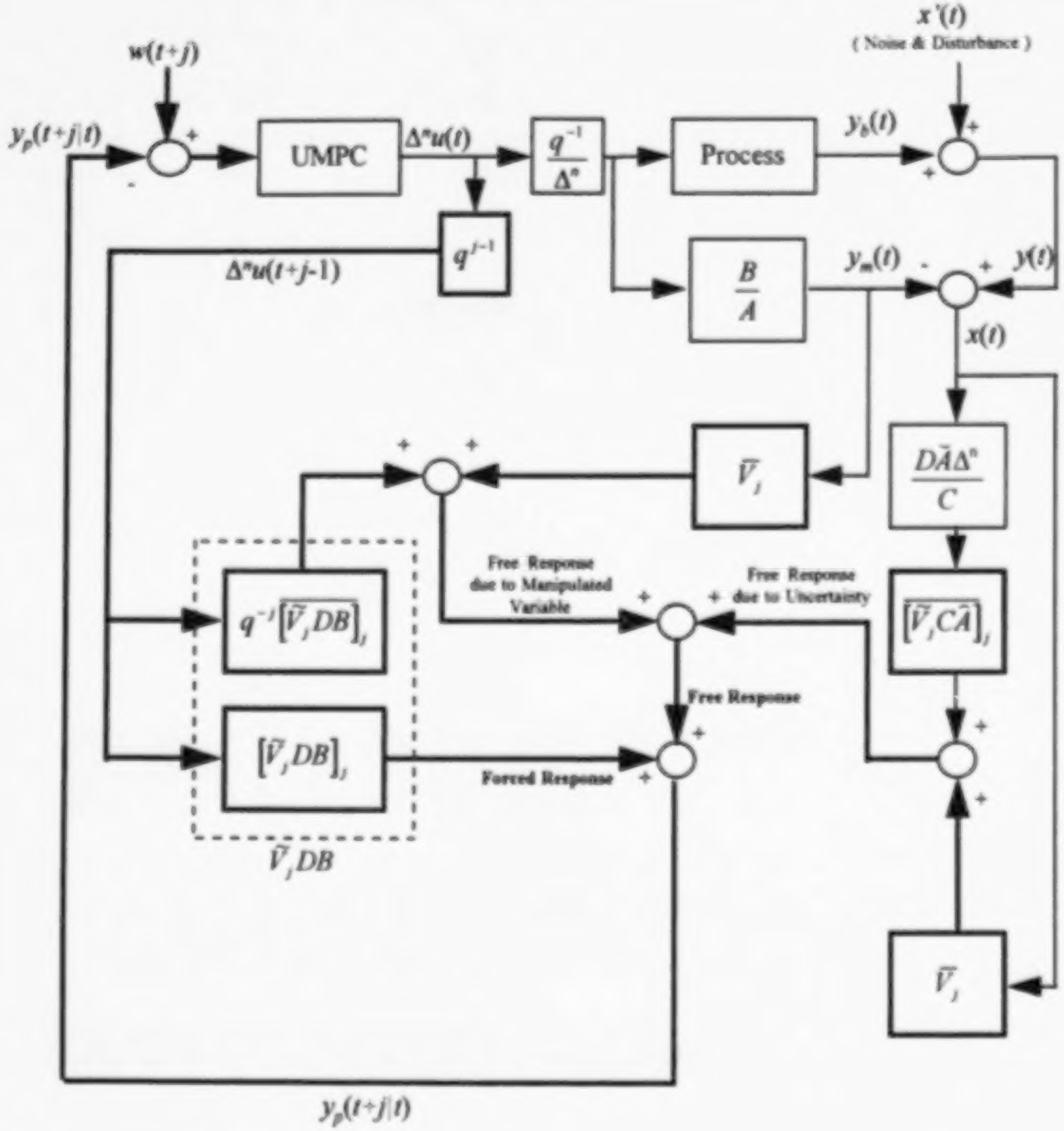

$x'(t)$
(Noise & Disturbance)
$w(t+j)$
$y_p(t+j|t)$
UMPC
$\Delta^n u(t)$
$\frac{q^{-1}}{\Delta^n}$
Process
$y_b(t)$
q^{j-1}
$\Delta^n u(t+j-1)$
$\frac{B}{A}$
$y_m(t)$
$y(t)$
$x(t)$
$\frac{D\tilde{A}\Delta^n}{C}$
$e(t)$
$q^{-j}\overline{[\tilde{V}_j DB]}_j$
$[\tilde{V}_j DB]_j$
$\overline{[\tilde{V}_j C\tilde{A}]}_j$
Free Response
Forced Response
$\tilde{V}_j DB$
$\tilde{V}_j$
$y_p(t+j|t)$

Notes:

1) The filters and signals with the index j are vectors and are drawn with thicker lines; j varies from 1 to N_2.

2) The UMPC block does the calculations (optimization) necessary to produce the control vector that minimizes the predicted control errors.

Figure 4.1 Unified model predictive control (UMPC) with reduced Diophantine predictor (RDP)


x'(t)
(Noise & Disturbance)
w(t+j)
y_p(t+j|t)
+
-
UMPC
Δ^n u(t)
q^-1 / Δ^n
Process
y_b(t)
+
+
q^(j-1)
Δ^n u(t+j-1)
B / A
y_m(t)
-
+
y(t)
x(t)
+
+
Ṽ_j
DÃΔ^n / C
Free Response due to Manipulated Variable
Free Response due to Uncertainty
+
+
q^-j [Ṽ_j DB]_j
[Ṽ_j CÃ]_j
Free Response
+
[Ṽ_j DB]_j
Forced Response
+
+
+
Ṽ_j DB
Ṽ_j
y_p(t+j|t)

Notes:

1) The filters and signals with the index j are vectors and are drawn with thicker lines; j varies from 1 to N_2.

2) The UMPC block does the calculations (optimization) necessary to produce the control vector that minimizes the predicted control errors.

Figure 4.2 Analysis of the RDP to separate free responses corresponding to the manipulated variable and the uncertainty

Chapter 5

Generalized Cost Function for Long Range Predictive Control

A generalized cost function for Long Range Predictive Control (LRPC) is introduced which allows optimization of the sum of squares of arbitrarily selected tracking errors within the output horizon (i.e. it allows for a *sparse* output horizon) and hence reduces the computational load associated with the conventional *full* output horizon . Regular spacing, d_1, is suggested as a convenient way to specify the sparse output horizon but arbitrary point-by-point specification is also possible.

For regularly-spaced highly sparse output horizons the tracking error *summation* term in the cost function is replaced by a *numerical integration formula* in order to eliminate the deterioration of performance caused by removing a large number of tracking errors from the minimization process.

The UMPC cost function also allows for a sparse control horizon. *Independent* control moves (future control moves which are treated as independent variables in the optimization) may be defined arbitrarily anywhere in the output horizon as opposed to the contiguous control horizon of classical LRPC. In general the *non-independent* control moves (future control moves which are arbitrarily defined in terms of independent control moves) may be a function of the independent control moves. A sparse control horizon, with the non-independent control moves set to zero, gives slower control than that obtained using a corresponding full control horizon. Two special cases of sparse control horizons are presented in detail. A control horizon of N_u independent control moves with a spacing, d_2, between the first and the rest of independent control moves gives the effect of a continuous (fractional) control horizon between 1 and N_u as d_2 is increased. Similarly a control horizon of N_u independent control moves with a spacing, d_3, between the last and the rest of independent control moves gives the effect of a fractional control horizon between N_u -1 and N_u . An effect equivalent to a fractional control horizon between two integer values can also be achieved by dividing the output horizon into two parts using d_4 as divider and employing a different control horizon for each of the two parts.

Simulation results are presented to illustrate the effect of various aspects of the newly introduced generalized cost function.

5.1 Introduction

The purpose of a long-range predictive controller is to find the current (at time = t) control action by considering the effect of current plus future control moves on the output over a future horizon bounded by times $t+N_1$ and $t+N_2$. Time $t+N_1$ is greater than or equal to the earliest time in the future at which the output is affected by the control move at time t, while N_2 is arbitrary but should be greater than N_1.

To date most of the popular long-range predictive controllers minimize the following type of quadratic cost functions for the optimal control calculation (Cutler, 1979, Clarke, 1987; McIntosh, 1988):

$$J = \sum_{j=1}^{N} \gamma(j)\left[y_p(t + N_1 + j - 1|t) - w(t + j)\right]^2 + \sum_{j=1}^{N_u} \lambda(j)[\Delta u(t + j - 1)]^2 \tag{5.1}$$

where the terms are defined in Chapters 2.

Note that for the above cost function:

$$N = N_2 - N_1 + 1 \tag{5.2}$$

The prediction $y_p(t + j|t)$ in the above cost function is usually given by an optimal predictor and may be described by the following general equation:

$$y_p(t + j|t) = \widetilde{G}_j \Delta u(t + j - 1) + f(t + j) \tag{5.3}$$

$$\widetilde{G}_j = g_0 + g_1 q^{-1} + \ldots + g_{j-1} q^{-j+1} \tag{5.4}$$

The prediction, $y_p(t+j|t)$, consists of two parts. The first term on the RHS of the predictor is the response due to the future input moves and is known as the *forced response*. The second term on the RHS of the predictor, $f(t+j)$, is called the *free-response*. It is the response due to the past inputs $u(.)$ and disturbances up to time t.

The control law is given as (McIntosh, 1988):

$$\mathbf{u} = \left[\mathbf{G}^{\mathrm{T}}\Gamma\mathbf{G} + \Lambda\right]^{-1}\mathbf{G}^{\mathrm{T}}\Gamma(\mathbf{w} - \mathbf{f}) \tag{5.5}$$

where the terms are the same as used in Chapter 2. The dynamic matrix **G** is given as:

$$\mathbf{G} = \begin{bmatrix} g_{N_1-1} & \cdots & \cdots & g_0 & \cdots & 0 \\ g_{N_1} & \cdots & g_2 & g_1 & \ddots & 0 \\ \vdots & \vdots & \vdots & & \cdots & g_0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ g_{N_2-1} & g_{N_2-2} & \cdots & \cdots & \cdots & g_{N_2-N_u} \end{bmatrix}_{(N_2-N_1+1)\times N_u} \tag{5.6}$$

for $N_u = N_2$

$$\mathbf{G} = \begin{bmatrix} g_{N_1-1} & \cdots & g_0 & 0 & \cdots & 0 \\ g_{N_1} & \cdots & g_1 & g_0 & \ddots & 0 \\ \vdots & \vdots & \vdots & & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ g_{N_2-1} & g_{N_2-2} & \cdots & \cdots & \cdots & g_0 \end{bmatrix}_{(N_2-N_1+1)\times N_2} \tag{5.7}$$

Figure-5.1 describes the underlying concepts of long range prediction and output and control horizons for a conventional LRPC (with $N_1 = 1$, $N_2 = 9$ and $N_u = 5$ for illustration purposes).

In the following development the boundaries of the output horizon are defined exactly in the way they are employed in the conventional LRPC, i.e. N_1 and N_2 or for ease of notation 1 and N. However the control horizon is understood to be the number of control moves which are not arbitrarily specified (i.e. the *independent* control moves). Note that in the conventional LRPC formulation the control horizon, N_u, also defines the future time instant beyond which the control moves are arbitrarily set to zero. This *positional* (in time dimension) aspect of the conventional control horizon definition *is not* followed in the following derivations. For example a control horizon of 3 will specify that three independent control moves are involved. It does not necessarily mean however that the control horizon spans the interval t to t+2.

5.2 Output Horizon Restructuring

LRPC with the conventional output horizon (which includes the future predictions at every consecutive time instant from $t+N_1$ to $t+N_2$) requires substantial computation and memory. This is especially true for MIMO systems. The following sections present an easy way to restructure the output horizon so that fewer output predictions are used in the control law calculation.

5.2.1 Sparse Output Horizon

The cost function of equation (5.1) may be written as a sum of two terms:

$$J = J_e + J_c \tag{5.8}$$

$$J_e = \text{ tracking error cost } = \sum_{j=1}^{N} \gamma(j)\left[y_p(t + N_1 + j - 1|t) - w(t + j)\right]^2 \tag{5.9}$$

$$J_c = \text{ control cost } = \sum_{j=1}^{N_u} \lambda(j)[\Delta u(t + j - 1)]^2 \tag{5.10}$$

A long output horizon is often needed to get a stable closed-loop response in the presence of unmodeled or unusual plant dynamics. However a large output horizon means more computation and computer storage since the tracking error cost J_e will have more terms in the summation. One way to reduce the computational load without sacrificing the benefit of a large output horizon is to use a sparse output horizon in J_e . In other words the cost function will minimize the sum of squares of tracking errors at *selected* points rather than all of the points in the output horizon. In this case J_e is defined as:

$$J_e = \sum_{j=\exists(N_1 \ldots N_2)} \gamma(j)\left[y_p(t+j|t) - w(t+j)\right]^2 \qquad (5.11)$$

where ∃ is the existential quantifier and j=∃(N_1...N_2) indicates that the summation may involve any arbitrary time instants in the output horizon. Any number of time instants may be included in the above summations. However the time instants $t+N_1$ and $t+N_2$ must be included in the summation to preserve the specific output horizon boundaries.

The above error cost is very general. A more easily specified J_e involves summation of terms at equally spaced time instants. (i.e. regular spacing). For this special case the cost function may be specified as:

$$J_e = \sum_{j=1}^{N} \gamma(j)\left[y_p(t+N_1+[j-1)][d_1+1]|t) - w(t+N_1+[j-1)][d_1+1])\right]^2 \qquad (5.12)$$

where

N = the number of tracking errors in the summation.

d_1 = the spacing between predictions.

Note that if N is prespecified then N_2 can not be specified independently, rather it is given as:

$$N_2 = N_1 + (N-1)(d_1+1) \qquad (5.13)$$

On the other hand if N_2 is specified first and (N_2 -N_1) is divisible by (d_1+1) then N is fixed to the following value:

$$N = \frac{N_2 - N_1}{d_1 + 1} + 1 \qquad (5.14)$$

Note that for d_1 = 0 (i.e. for no spacing) (5.14) reduces to (5.2) giving the same N as in the conventional LRPC output horizon.

The above idea of a regularly-spaced, sparse output horizon is illustrated by Figure-5.2 which assumes d_1 = 3 and N = 4.

Another way to structure the summation in J_e is to divide the output horizon in two or more parts and use $d_1 = 0$ for the first part of the output horizon and for the rest of the output horizon use $d_1 > 0$ in an increasing trend. This works well because the variability of the tracking errors decreases along the output horizon.

Incorporation of the sparse output horizon in the control algorithm is very simple. Matrix **G**, and vectors **f** and **w** are modified by retaining only those rows which correspond to the tracking errors included in the error cost J_e. For regular spacing d_1 these are given as:

$$\mathbf{G} = \begin{bmatrix} g_{N_1-1} & \cdots & g_0 & 0 & \cdots & 0 \\ g_{N_1+d_1} & \cdots & \cdots & & \ddots & 0 \\ \vdots & \vdots & \vdots & & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ g_{N_2-1} & g_{N_2-2} & \cdots & \cdots & \cdots & g_0 \end{bmatrix}_{N \times N_2} \tag{5.15}$$

$$\mathbf{f} = \begin{bmatrix} f(t+N_1) \\ f(t+N_1+d_1+1) \\ f(t+N_1+2[d_1+1]) \\ \vdots \\ f(t+N_2) \end{bmatrix}_{N \times 1} \quad \text{and} \quad \mathbf{w} = \begin{bmatrix} w(t+N_1) \\ w(t+N_1+d_1+1) \\ w(t+N_1+2[d_1+1]) \\ \vdots \\ w(t+N_2) \end{bmatrix}_{N \times 1} \tag{5.16}$$

$$\mathbf{G} = \begin{bmatrix} \mathbf{g}_1 & \mathbf{g}_2 & \cdots & \mathbf{g}_{N_2} \end{bmatrix}_{N \times N_2} \quad \text{where the } \mathbf{g}_i\text{'s are columns of size } N \tag{5.17}$$

Effect of d_1:

Simulation results have shown that for fixed N_2, the control performance can usually be maintained with d_1 greater than zero but the results depend on the process model and the selected control interval. This means that fewer free responses have to be calculated. The dimensions of the various vectors and matrices, involved in the control calculation, are also reduced. This reduces the memory storage requirements and the required computational load. Note that the computational savings obtained due to the reduced number of free responses is valid for both the adaptive and the non-adaptive implementations.

The computational and storage savings due to d_1 spacing is of particular interest for MIMO case where the number of free responses to be calculated on-line is much greater. If the predictions are calculated using recursive equations then the same number of predictions (e.g. N_2 values) are calculated even though a subset is used in subsequent calculations. However, for example for a 4×4 MIMO system with $N_2 = 21$, there are 16 sets of free responses each of which has 21 points. Therefore a total of 336 free response points must be calculated on-line at each control interval for a regular non-sparse output horizon (i.e. for $d_1 = 0$). However, substantial computational and storage savings are achieved if a sparse output horizon, e.g. with $d_1 = 4$, is employed. In this case there will be 16 sets of free responses each of which contains only 5 (instead of 21) points. Consequently, instead of 336, only 80 free responses have to be calculated. Besides free responses, the sizes of the various vectors and matrices are also reduced.

Note that in practice a single d_1 may not be appropriate for all (16 in the above example) sets of predictions because of different dynamics of various transfer functions of a MIMO transfer function matrix. More than one d_1 can be used to deal with this situation.
For d_1 >>0, the control performance deteriorates. This is understandable, since in this case the generalized cost function minimizes the tracking error at fewer time instants in the output horizon.

5.2.2 Integral Error Cost

For the regular spacing case the performance of the controller deteriorates as the spacing d_1 increases. An interesting solution to this problem is to apply appropriate weightings to the tracking errors in the *summation* of J_e to mimic an *integration* of the squares tracking errors. These weightings are obtained using Simpson's rule or any higher order numerical integration methods (e.g. Newton-Cotes formulae) and are incorporated in the control algorithm by modifying the $\gamma(j)$'s of Γ in the control law.

5.3 Generalized Control Horizon

5.3.1 Planned control Policy (PCP)

The cost function (5.1) can be minimized with respect to all N_2 possible control moves. However it is often advantageous to impose certain assumptions on these control moves. This concept of arbitrarily constraining some of the future control moves is referred to as a *Planned Control Policy* or PCP. The concept of PCP is not new and in fact a particular PCP which sets all $\Delta u(.)=0$ after N_u, has been employed in the above derivation and in many well known LRPC's (e.g. DMC, GPC) because it reduces the matrix **G** of (5.17) to:

$$\mathbf{G} = \begin{bmatrix} \mathbf{g}_1 & \mathbf{g}_2 & \cdots & \mathbf{g}_{N_u} \end{bmatrix}_{N \times N_u} \tag{5.18}$$

In this section the concept of a PCP is defined much more generally and presented more formally.

The PCP adopted in this study classifies the control moves to be calculated as *independent* and *dependent* control moves. Only the *independent* control moves are used in the minimization of the cost function or generated in the calculated control vector. The *dependent* control moves may be arbitrarily fixed to a constant (e.g. zero) or defined in terms of the independent control moves. The control move at time t must be included as an independent move to give the independent control for the current time (t). The control cost function, J_c, involves only the weighted sum of the squares of the *independent* control moves. De Keyser (1991) suggests similar methods of structuring of the control law.

A general control cost is defined as:

$$J_c = \sum_{j=\exists(1 \ldots N_2)} \lambda(j)[\Delta u(t+j-1)]^2 \tag{5.19}$$

Again this control cost is too general. More easily specifiable special cases exist. In the following sections two useful control costs, are introduced. These control costs assume that the number of *independent* control moves N_u and all the *dependent* control moves are arbitrarily set to zero. The two sparse control horizons can be obtained from a conventional contiguous control horizon of size N_u by inserting the following spacing:

1. Spacing, d_2, between the first (current) control move and the rest of the N_u -1 independent contiguous control moves to produce the effect of a control horizon between 1 and N_u **or,**
2. Spacing, d_3, between the first N_u -1 contiguous control moves and the last control move to produce the effect of a control horizon between N_u and N_u-1.

5.3.2 The d_2 Spacing

The following control cost provides a means to incorporate the above defined d_2 spacing (i.e. the spacing between the first (current) control move and the rest of N_u -1 independent contiguous control moves) in the conventional control horizon:

$$J_c = \lambda(1)[\Delta u(t)]^2 + \sum_{j=2}^{N_u} \lambda(j)[\Delta u(t+j-1+d_2)]^2 \tag{5.20}$$

where

$N_u =$ number of independent control moves.

$d_2 =$ spacing between the first and the rest of the independent control moves.

The independent control moves specified in J_c must lie between t and $t+N_2-1$ which implies that d_2 should be constrained to observe the following condition:

$$d_2 \le N_2 - N_u \tag{5.21}$$

The dependent control moves, $\Delta u(.)$, are set to be zero which means that the control action, $u(.)$, remains at its previous value. The control vector is:

$$\mathbf{u} = \begin{bmatrix} \Delta u(t) \\ \hdashline \Delta u(t+1+d_2) \\ \Delta u(t+2+d_2) \\ \vdots \\ \Delta u(t+N_u-1+d_2) \end{bmatrix}_{N_u \times 1} \tag{5.22}$$

The matrix **G** of (5.17) is modified to incorporate d_2 by removing the columns corresponding to the dependent control moves, i.e.

$$\mathbf{G} = \left[\mathbf{g}_1 \;\vdots\; \mathbf{g}_{2+d_2} \quad \cdots \quad \mathbf{g}_{N_u+d_2} \right]_{N \times N_u} \tag{5.23}$$

The above formulation gives $N_2 - N_u + 1$ discrete levels of control horizon interpolation between 1 and N_u. In other words the d_2 spacing produces the effect of a fractional control horizon between 1 and N_u. For large N_u the size and effect of d_2 will be smaller. In the extreme case of $N_u = N_2$, d_2 can not be greater than zero and only one control horizon (i.e. $N_u = N_2$) is available. Figure-5.3 gives a diagrammatic description of the generalized control horizon with d_2 spacing.

5.3.3 The d_3 Spacing

The d_2 spacing provides $N_2 - N_u + 1$ levels of apparent fractional control horizon between 1 and N_u (i.e. a span of N_u control moves). With d_3 spacing (i.e. the spacing between the first N_u -1 contiguous control moves and the last control move) the same number of levels of apparent fractional control horizon is obtained for a span of one control horizon (i.e. between N_u -1 and N_u). In other words d_3 spacing gives better resolution in terms of number of control horizon levels per control move. d_3 spacing is included in the conventional control horizon using the following control cost:

$$J_c = \left[\sum_{j=1}^{N_u-1} \lambda(j)[\Delta u(t+j-1)]^2 \right] + \lambda(N_u)[\Delta u(t+N_u-1+d_3)]^2 \tag{5.24}$$

where

$N_u =$ number of independent control moves.

$d_3 =$ spacing between the last two independent control moves.

Moreover the independent control moves specified in J_c must lie between t and $t+N_2-1$. In other words:

$$d_3 \le N_2 - N_u \tag{5.25}$$

Note that now the $N_2 - N_u + 1$ levels (of interpolated control horizon) are available over a single control move, a much better resolution for control horizon interpolation. The dependent control moves are set to be zero so that:

$$\mathbf{u} = \begin{bmatrix} \Delta u(t) \\ \Delta u(t+1) \\ \vdots \\ \Delta u(t+N_u-2) \\ \hdashline \Delta u(t+N_u-1+d_3) \end{bmatrix}_{N_u \times 1} \tag{5.26}$$

Note that dependent control moves between $\Delta u(t+N_u-2)$ and $\Delta u(t+N_u-1+d_3)$, which were arbitrarily assumed to be zero, are not included in vector the **u**. Similarly the dependent control moves after $\Delta u(t+N_u-1+d_3)$ do not appear in **u**.

Again the original matrix **G** , of (5.17), is modified to incorporate d_3 by removing the columns corresponding to the dependent control moves. In terms of equation **G** is given as:

$$\mathbf{G} = \begin{bmatrix} \mathbf{g}_1 & \mathbf{g}_2 & \cdots & \mathbf{g}_{N_u-1} & \vdots & \mathbf{g}_{N_u+d_3} \end{bmatrix}_{N \times N_u} \tag{5.27}$$

Note that the columns of the original **G** matrix, in (5.17), between $\mathbf{g}_{N_u-1}$ and $\mathbf{g}_{N_u+d_3}$ are not included in the above modified **G**. Incorporation of d_3 in the control horizon is demonstrated in Figure-5.4.

5.4 Multiple Control Horizons

The conventional cost function assumes a *single control horizon* for the entire output horizon. As already discussed, a more general (flexible) PCP (Planned Control Policy may involve the use of *two (or multiple) control horizons* in the control law.

The output horizon may be divided into two or more parts and a separate control horizon can be defined corresponding to each part of the output horizon. In the following development a smaller *control* horizon is used for the beginning part of the *output* horizon and a larger control horizon is employed for the later part. A tuning parameter d_4, the extent of the smaller control horizon, is used to divide the output horizon into two parts.

The first part, which uses a smaller control horizon, spans the time t to $t+d_4$ and the rest of the output horizon, d_4+1 to N_2, uses a larger control horizon. There are many possible combinations for selecting the two control horizons for the two parts of the output horizon. Two useful cases for control horizon selection are discussed in the following.

5.4.1 Control Horizons 1 and N_u

In this strategy control horizons of 1 and N_u are employed for the first and second parts of the output horizon respectively. The tuning parameter d_4, which takes values in the range 0 to N_2, determines the relative effect of the smaller control horizon which in this case is 1. For $d_4 = 0$ the control law uses a control horizon exactly equal to N_u. Whereas $d_4 = N_2$ gives a controller with control horizon of 1. As d_4 increases from 0 to N_2 the effect is equivalent to a continuous change in the control horizon from N_u to 1. The effect of d_4 in this arrangement(i.e. with control horizons 1 and N_u) is similar to that of d_2 mentioned earlier in section 5.3.2.

The above strategy can be implemented by setting the last N_u-1 columns of the first d_4 rows of the matrix **G** equal to zero. The modified **G** matrix is given as follows:

$$\mathbf{G} = \left[\begin{array}{c|ccccc} g_{N_1-1} & 0 & 0 & 0 & \cdots & 0 \\ g_{N_1} & 0 & 0 & 0 & \ddots & 0 \\ \vdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ g_{N_1+d_4-1} & 0 & 0 & 0 & \ddots & 0 \\ \hline g_{N_1+d_4} & g_{N_1+d_4-1} & \cdots & \cdots & \cdots & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ g_{N_2-1} & g_{N_2-2} & \cdots & \cdots & \cdots & g_{N_2-N_u} \end{array}\right]_{N\times N_u} \qquad (5.28)$$

Note that all elements of the upper right partition of **G** are set to be zero.

5.4.2 Control Horizons N_u-1 and N_u

The idea here is to provides the effect of a fractional control horizon between N_u-1 and N_u. This is accomplished by using a control horizon of N_u-1 for the first part of the output horizon and a control horizon of N_u for the remaining part. To incorporate these two consecutive control horizons in the control law the first d_4 rows of the last column of **G** are set equal to zero. The modified **G** is given as:

$$\mathbf{G} = \left[\begin{array}{ccccc|c} g_{N_1-1} & \cdots & \cdots & g_0 & \cdots & 0 \\ g_{N_1} & \cdots & g_2 & g_1 & \ddots & 0 \\ \vdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ g_{N_1+d_4-1} & \vdots & \vdots & \vdots & \ddots & 0 \\ \hline g_{N_1+d_4} & g_{N_1+d_4-1} & \cdots & \cdots & \cdots & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ g_{N_2-1} & g_{N_2-2} & \cdots & \cdots & \cdots & g_{N_2-N_u} \end{array}\right]_{N \times N_u} \tag{5.29}$$

Note that all elements of the upper right partition of **G** are set to be zero.

For $d_4 = 0$ the above modification gives a controller corresponding to control horizon exactly equal to N_u. and $d_4 = N_2$ gives a controller with control horizon of N_u-1. The role of d_4 in this method is similar to the interpolating effect of d_3.

5.5 Simulation Examples

The following sections present simulation examples were designed to highlight the following features and notions:

- A sparse output horizon may be used to reduce the dimensionality of the LRPC control law calculation.
- The use of numerical integration in place of simple summation in the cost function is helpful when large spacing in the output horizon leads to poor performance or even instability especially for larger control horizons and unmodeled dynamics.
- Spacing between the first and the rest of control moves in the control horizon.
- Spacing between the last and the rest of control moves in the control horizon.
- Use of two control horizons, 1 and N_u.
- Use of two control horizons, N_u-1 and N_u.

All simulations are based on the following 3rd order process :

$$\text{The } s\text{ - domain Process} = \frac{1}{(1+s)(1+3s)(1+5s)} \tag{5.30}$$

Using a sampling time of one second the following z-domain transfer function is obtained:

$$\text{The z - domain Process} = \frac{.00768z^{-1} + .02123z^{-2} + .00357z^{-3}}{1 - 1.90311z^{-1} + 1.1514z^{-2} - .2158z^{-3}} \tag{5.31}$$

The poles of the process in z-domain are 0.8187, 0.7165 and 0.3679, while the zeros are at -2.586 and -0.1798. Note that one of the zeros is outside the unit circle making the discretized process nonminimum phase (NMP). The gain of the process is 1.

The UMPC employed in the simulations uses a DMC noise model. The output was assumed to be noise-free, however a sustained step disturbance of magnitude 0.5 is added to the output at time instant = 50. The following parameters are common to all simulations: N_1=1, and λ=0 . In the following simulations perfect modeling is used, except for Examples 1 and 2 where the following Reduced Order Model is employed:

$$\text{Reduced Order Model} = \frac{0.12z^{-1} + 0.023z^{-2} 0.065z^{-3}}{1 - .91z^{-1}} \tag{5.32}$$

5.5.1 Example-1: Sparse Output Horizon

This simulation uses N_2=41 and N_u=2. Figure-5.5 shows that a spacing of 4 (i.e. d_1 = 4) in the output horizon does not deteriorate the control performance at all. The dimensions of the **G** matrix is 41×2 for d_1 = 0, whereas it is only 9×2 for the sparse output horizon case with d_1 = 4. Moreover 41 free responses need to be calculated for the regular output horizon case, while in case of the sparse output horizon only 9 free responses are required.

5.5.2 Example-2: Sparse Output Horizon with Integral Cost

This example demonstrates the following two facts:

I. A large value of d_1 destabilizes the system response

II. An integral cost can be employed to counter the destabilizing effect of larger d_1

In order to demonstrate the destabilizing effect of large d_1 a highly sparse output horizon with a spacing of $d_1 = 9$ has been used in this simulation. Output and control horizons are kept same as in Example-1 (i.e. N_2=41 and N_u=2). Figure-5.6 shows that an integral cost gives stable response almost identical to that in Example-1. In the absence of the integral cost an oscillatory (ringing) response results. Consequently the use of an integral cost allows larger spacing in the output horizon without deterioration of the closed-loop system response.

5.5.3 Example-3: Interpolating Effect of d_2

As indicated in section 5.3.1 the parameter d_2 can be used to interpolate between control horizons 1 and N_u . In this example the interpolating effect of d_2 is demonstrated by simulation. In order to appreciate d_2-interpolation, the responses corresponding to control horizons 1, 2 and 3 are given in Figure-5.7 where d_2 is set zero. In Figure-5.8 responses corresponding to three values of d_2 (7, 3 and 1) are plotted for a control horizon, N_u, of 3. The response for $d_2 = 7$ is almost equal to that for $N_u = 1$. The $d_2 = 3$ case gives control performance equivalent to an interpolation between $N_u = 1$ and $N_u = 2$. Setting $d_2 = 1$ gives an effect of a control horizon between $N_u = 2$ and $N_u = 3$.

5.5.4 Example-4: Interpolating Effect of d_3

This simulation example shows the interpolating effect of d_3 . A control horizon of 3 is used in Figure-5.9 which contains responses corresponding to three values of d_3 . All of these three responses give the effect of a control horizon between $N_u = 2$ and $N_u = 3$. The $d_3 = 7$ case is almost identical to $N_u = 2$ and $d_3 = 1$ gives a response that is similar to $N_u = 3$. The response corresponding to $d_3 = 3$ lies about half way between $N_u = 2$ and $N_u = 3$. Figure-5.10 illustrates the interpolating effect of d_3 between $N_u = 1$ and $N_u = 2$.

5.5.5 Example-5: Effect of Multiple Control Horizons

The proposition that more than one control horizons can be used to give the effect of an interpolated control horizon is illustrated by this example. Figure-5.11 shows the interpolating effect of d_4 between $N_u = 1$ and $N_u = 3$. Increasing d_4 shifts the response from $N_u = 3$ towards $N_u = 1$. Figure-5.12 demonstrates d_4-interpolation between $N_u = 1$ and $N_u = 3$.

5.6 Conclusions

The optimization criterion for LRPC's is viewed in a broader perspective. A generalized cost function is introduced which allows the use of a sparse output horizon, spacing in the control horizon or multiple control horizons. The specific results and conclusions of the current work are as follows:

- Minimization of the sum of *selected* tracking error squares in the output horizon yields results which compare well with the results when the sum of all of the tracking error squares is minimized.
- In some cases the performance of the selected tracking error squares minimization controller deteriorates if the number of selected tracking errors is below some minimum. Performance deterioration in case of a controller based on the minimization of very low number of selected tracking error squares may be avoided by using a numerical integration than a simple summation in the cost function.
- Spacing after the first independent control move produces an effect equivalent to using control horizons between 1 and N_u.
- Spacing before the last independent control move produces an effect equivalent to using control horizons between N_u-1 and N_u.
- Multiple control horizons provide the effect of control horizon $1 \leq N_u \leq. N_{umax}$

References:

Clarke, D.W., Mohtadi, C. and Tuffs, P.S., " Generalized Predictive Control - Part I the Basic Algorithm ", Automatica, Vol. 23, No. 2, pp.137-148, 1987a.

Clarke, D.W., Mohtadi, C. and Tuffs, P.S., " Generalized Predictive Control - Part II Extensions and Interpretations ", Automatica, Vol. 23, No. 2, pp. 149-160, 1987b.

Cutler, C.R., "Dynamic Matrix Control - A Computer Control Algorithm ", AIChE National Meeting, Houston, TX, 1979.

De Keyser, M.C., "Basic Principles of Model Based Predictive Control", Proceedings of the First European Control Conference, pp. 1753-1758, 1991.

McIntosh, A.R., "Performance and Tuning of Adaptive Generalized Predictive Control ", M.Sc. thesis, University of Alberta, 1988.

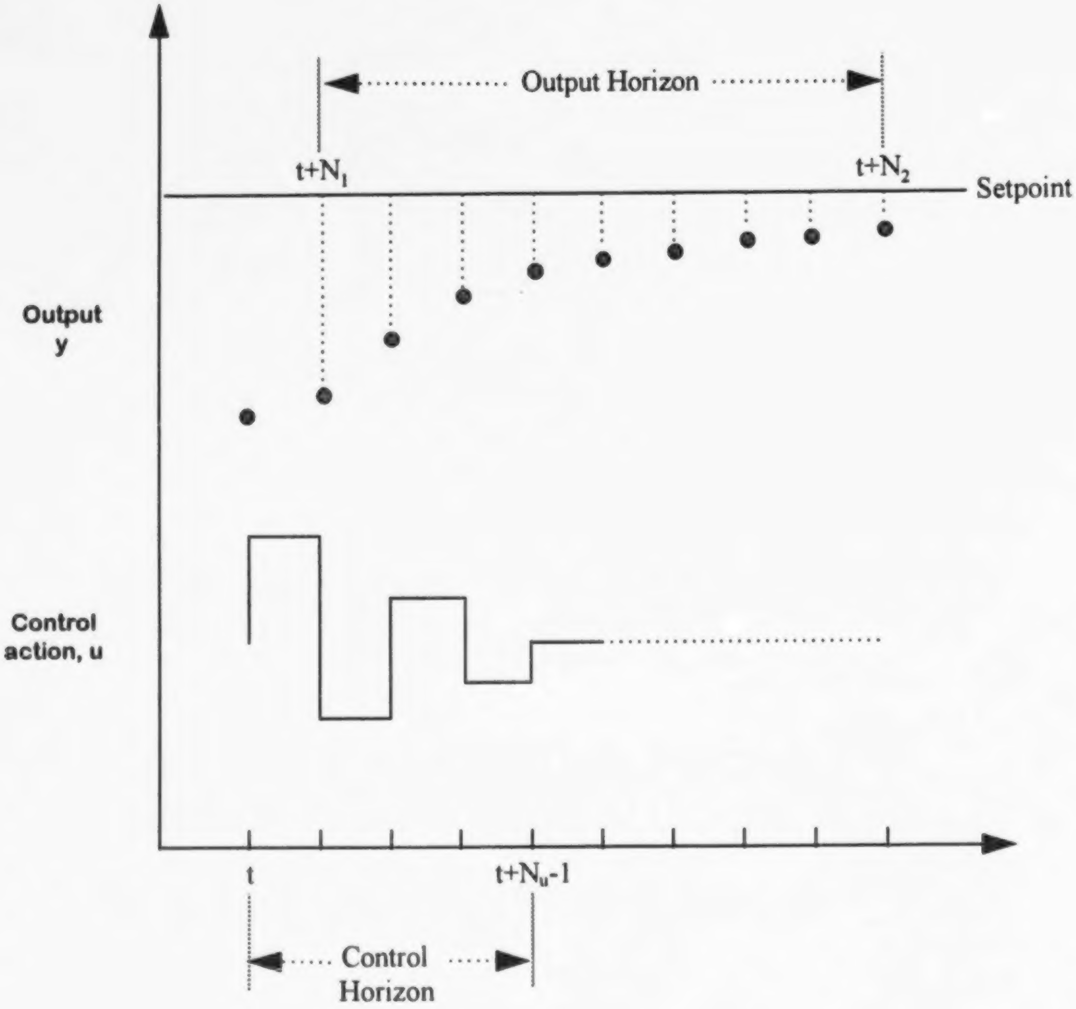

Output Horizon
t+N$_1$
t+N$_2$
Setpoint
Output
y
Control
action, u
t
t+N$_u$-1
Control
Horizon

Figure 5.1 Concept of a conventional long-range predictive controller

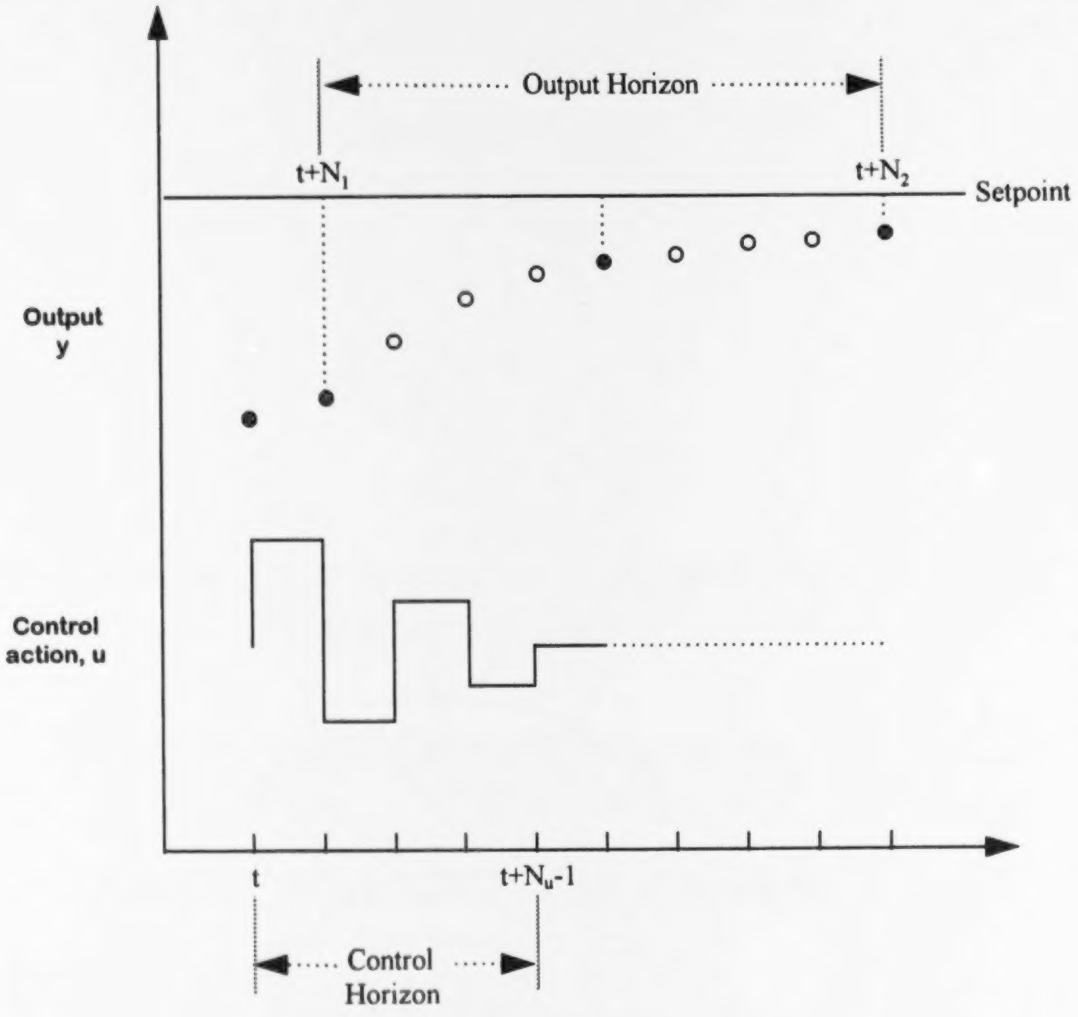

Output Horizon
t+N1
t+N2
Setpoint
Output
y
Control
action, u
t
t+Nu-1
Control
Horizon

Figure 5.2 Long-range predictive controller with sparse output horizon (d_1 = 3)

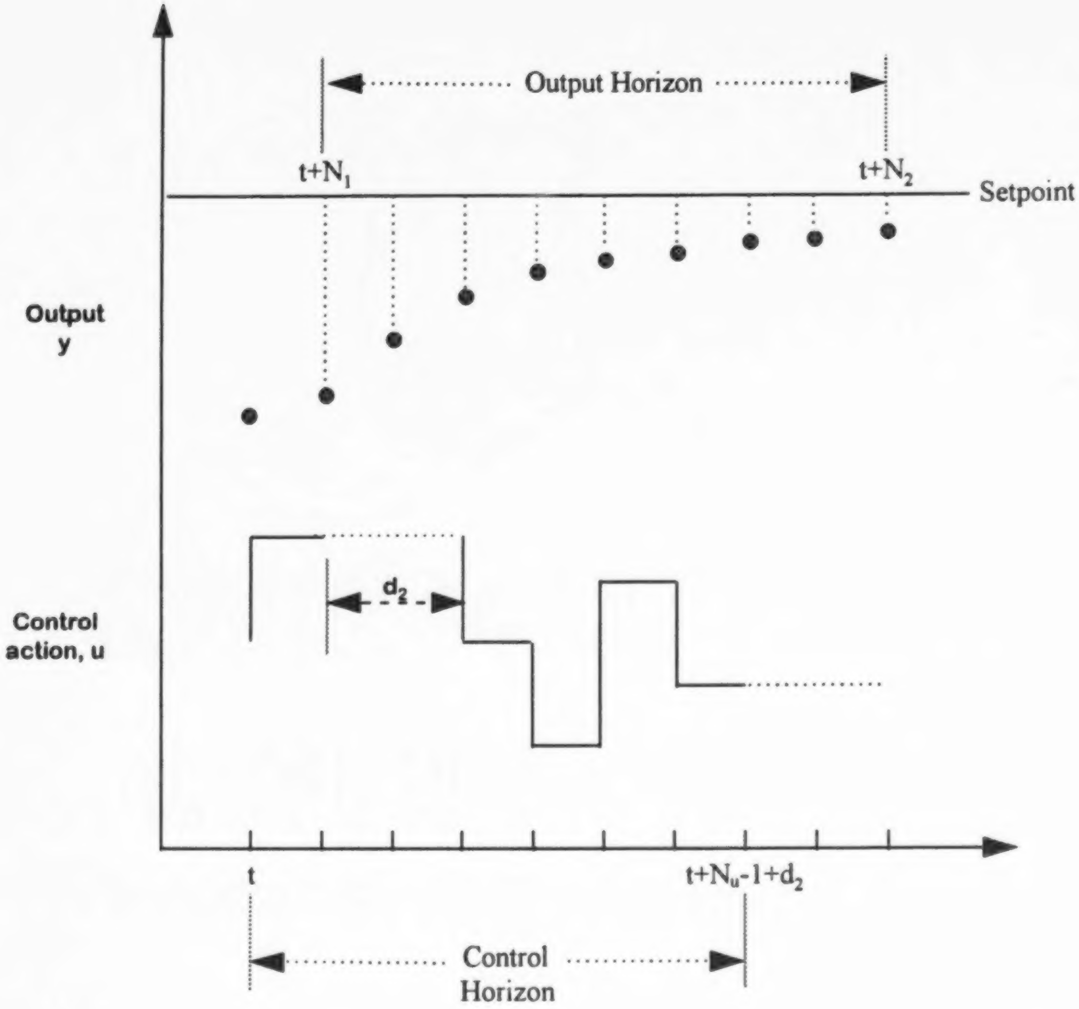

Output Horizon
t+N$_1$
t+N$_2$
Setpoint
Output y
Control action, u
d$_2$
t
t+N$_u$-1+d$_2$
Control Horizon

Figure 5.3 Long-range predictive controller with generalized control horizon, $d_2 = 2$

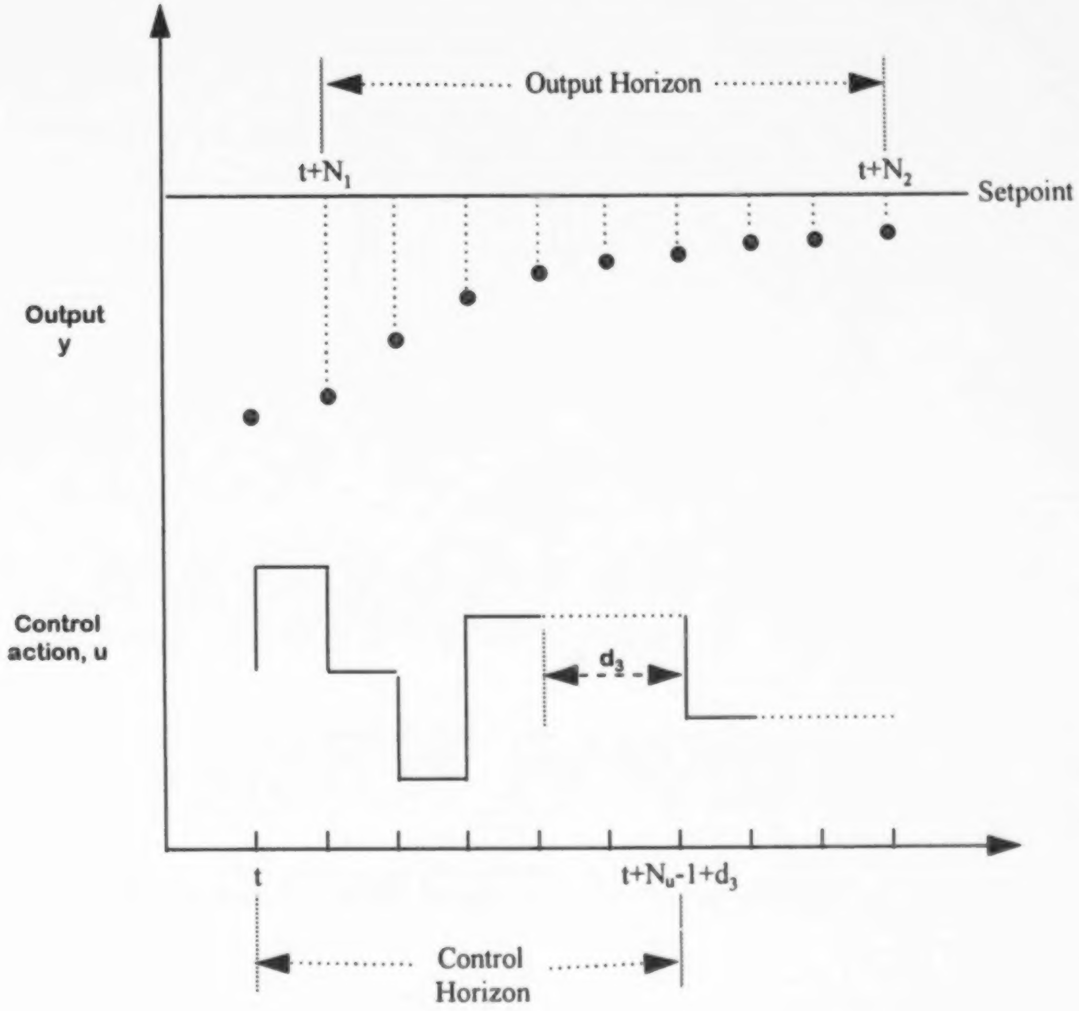

Output Horizon
t+N_1
t+N_2
Setpoint
Output
y
Control
action, u
d_3
t
t+N_u-1+d_3
Control
Horizon

Figure 5.4 Long-range predictive controller with generalized control horizon, $d_3 = 2$

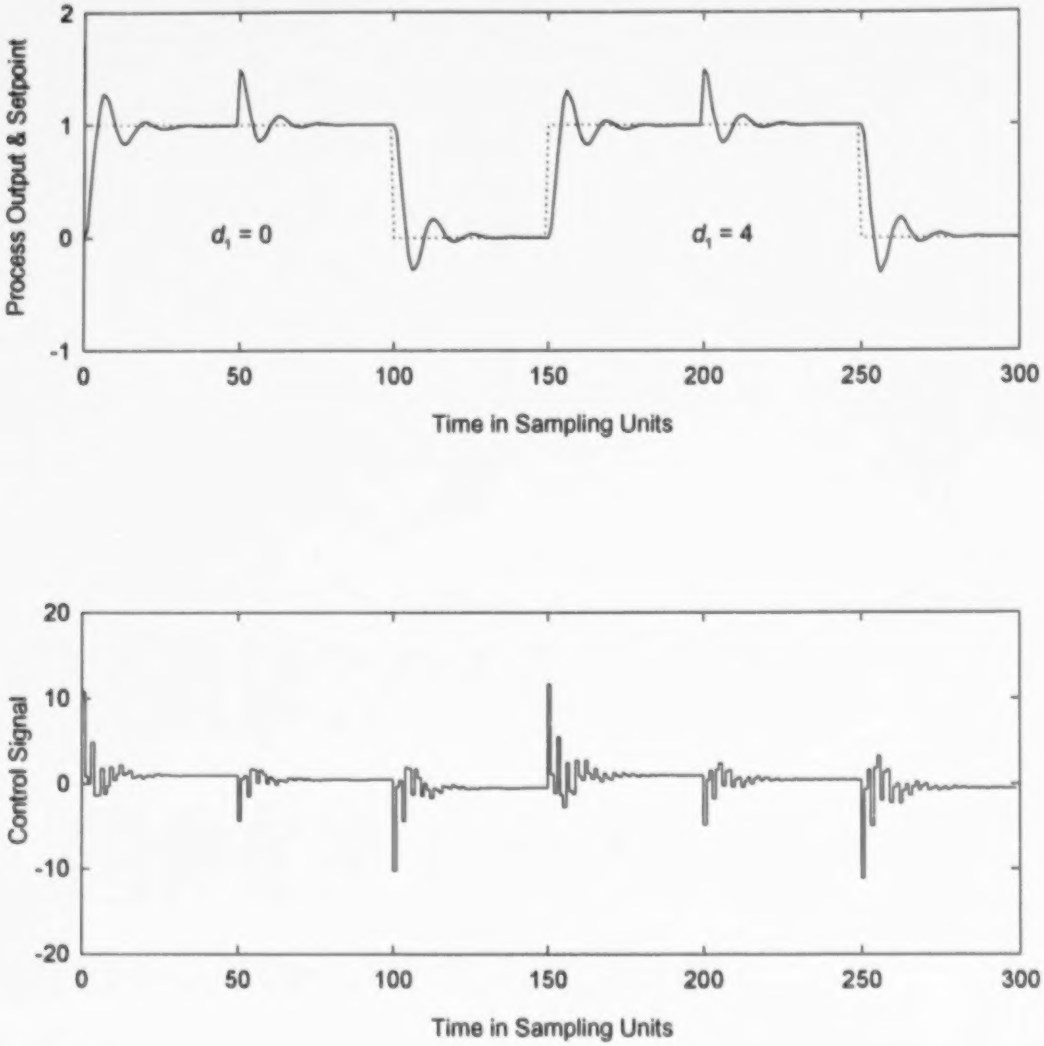


Figure 5.5 LRPC with regular and sparse output horizon

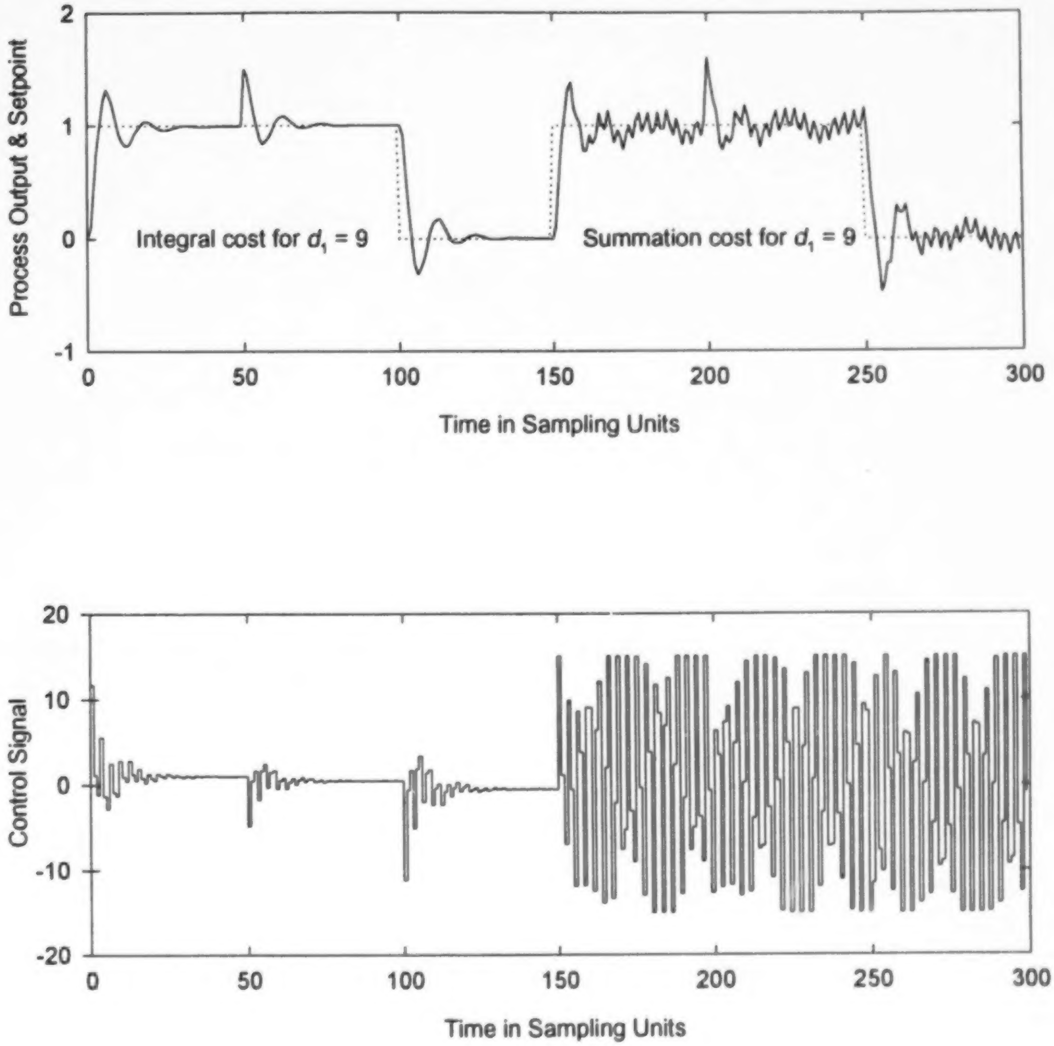


Figure 5.6 Integral and summation costs for LRPC with sparse output horizon

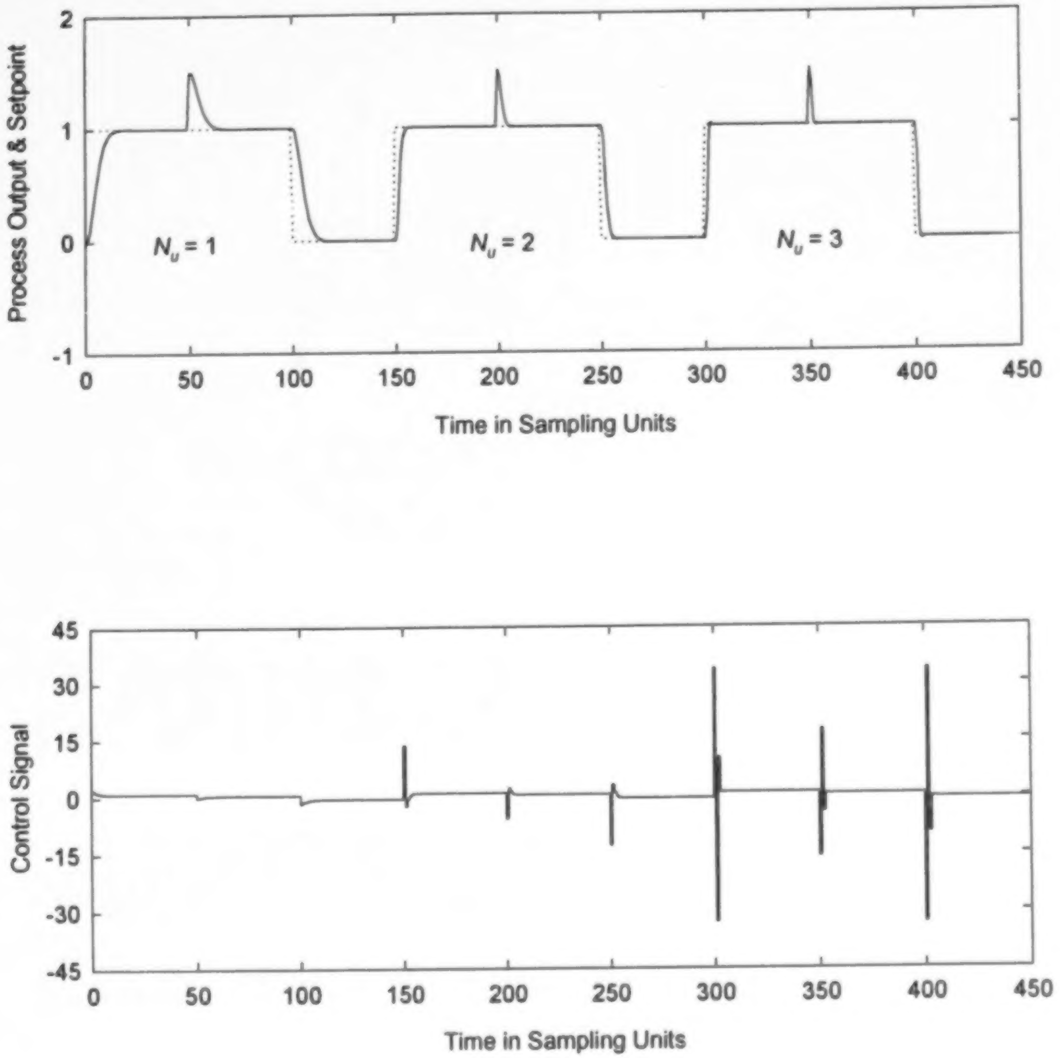


Figure 5.7 LRPC for control horizons 1, 2, and 3

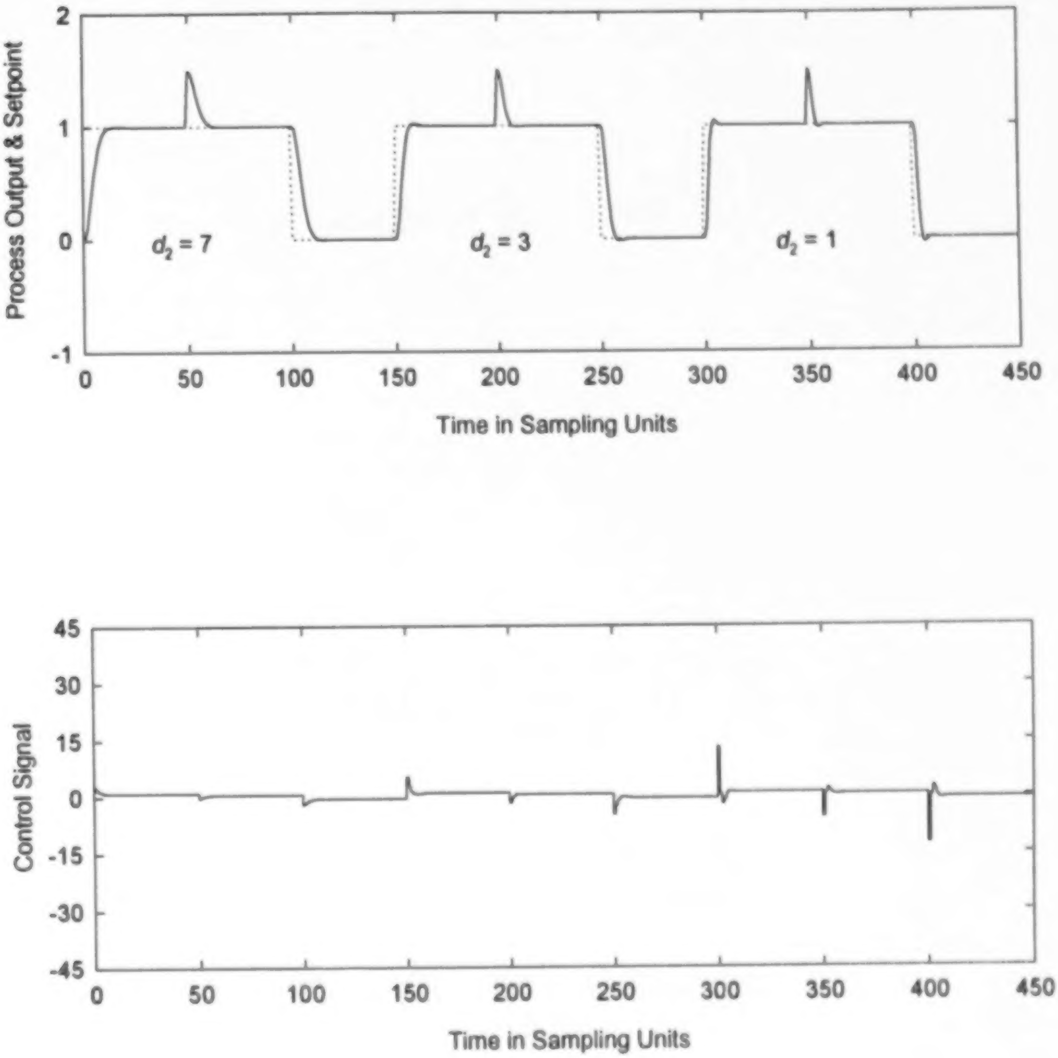
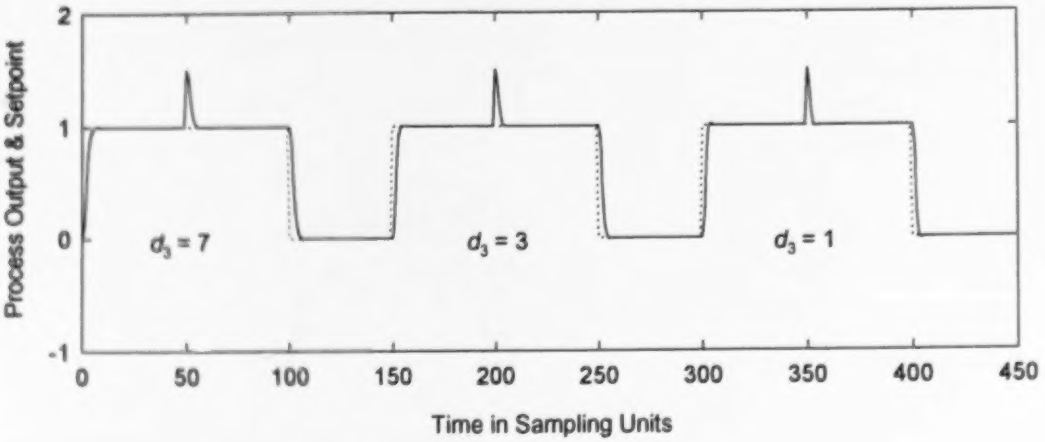


Figure 5.8 Interpolating effect of d_2



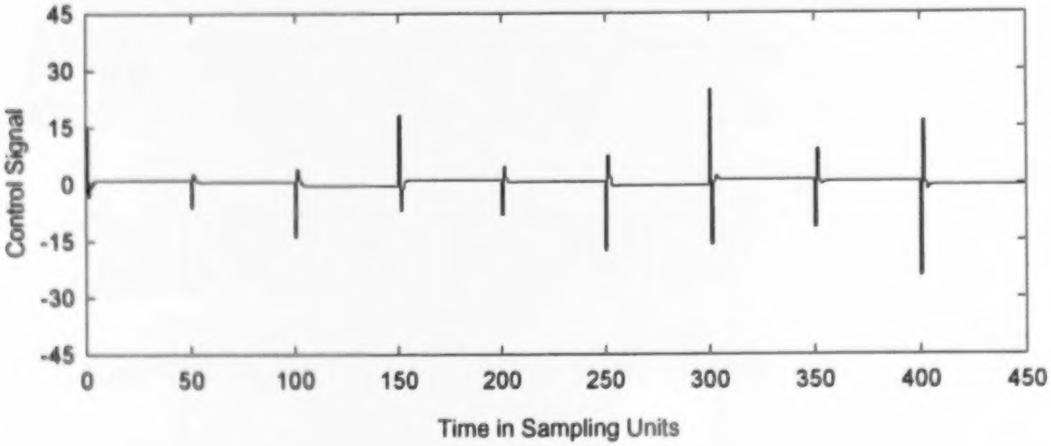


Figure 5.9 Interpolating effect of d_3 for $N_u = 3$

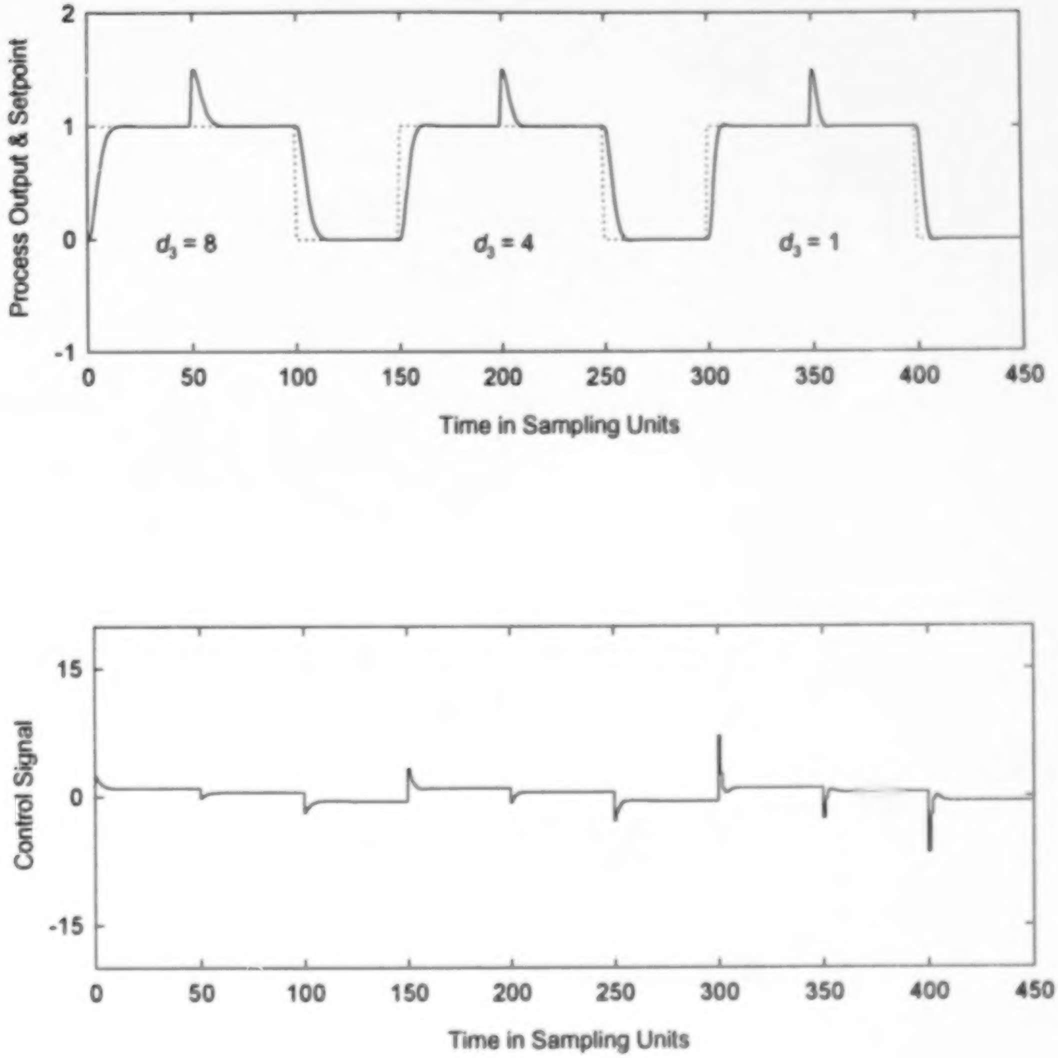
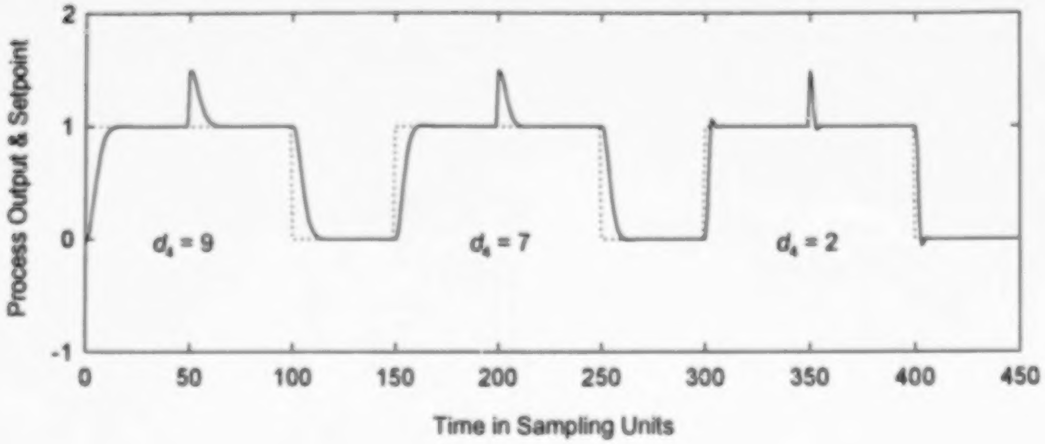


Figure 5.10 Interpolating effect of d_3 for $N_u = 2$



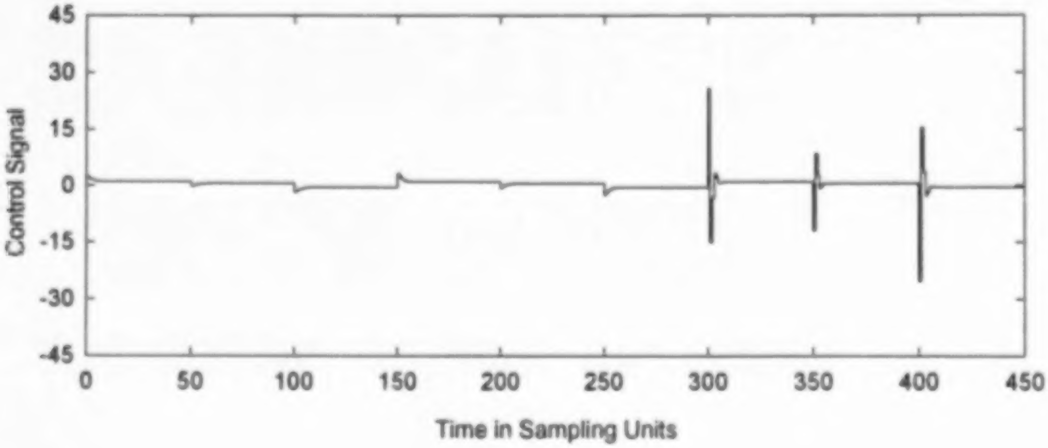


Figure 5.11 Multiple control horizons, N_u = 1 and N_u = 3

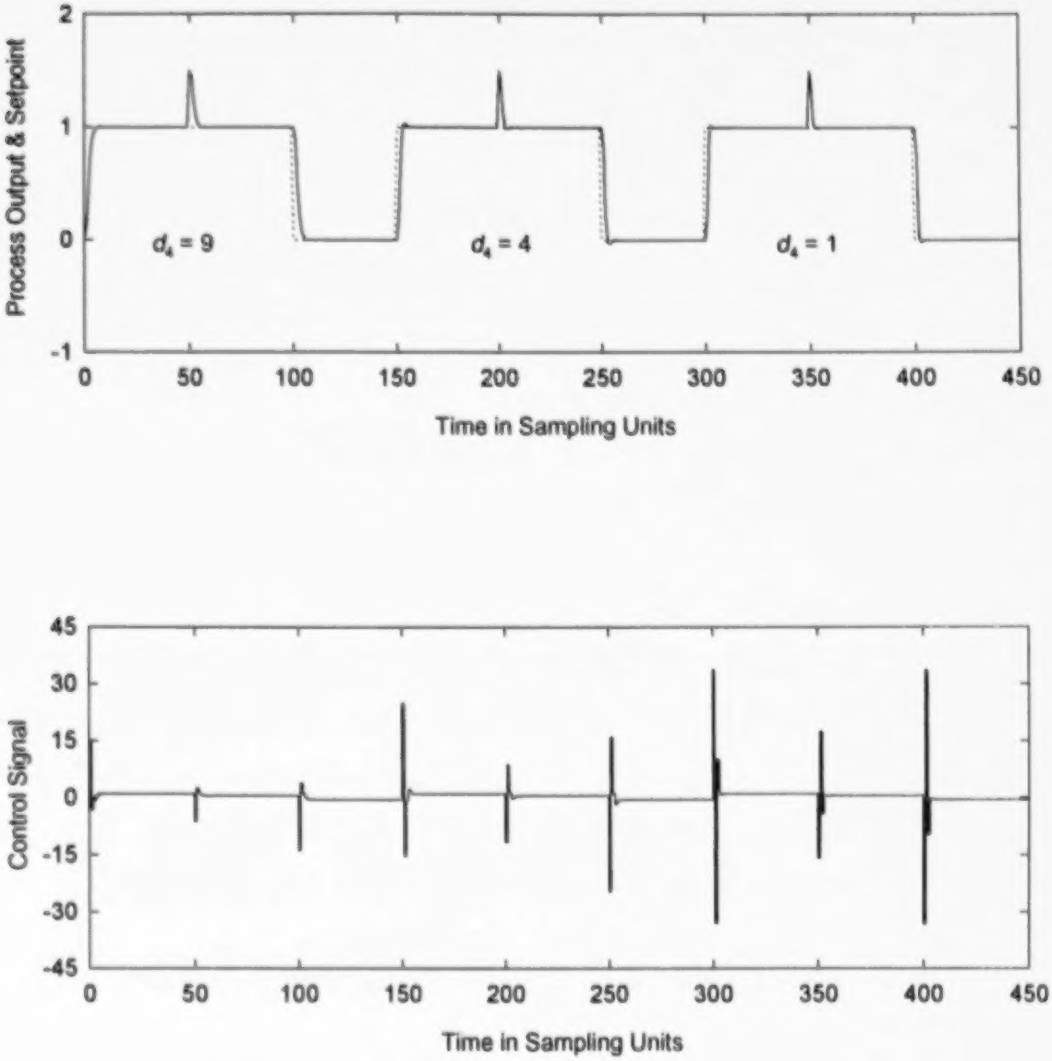


Figure 5.12 Multiple control horizons, $N_u = 2$ and $N_u = 3$

Chapter 6

Unified Model Predictive Control with Steady State Error Weighting

Unified Model Predictive Control (UMPC) is formulated using a generalized noise/disturbance model such that most linear discrete Long Range Predictive Controllers (LRPC's), e.g. GPC and DMC, are included as special or limiting cases. Steady state output control is added so that UMPC can be used with a smaller output prediction horizon N_2 and tuned by specifying a normalized weighting factor, $0 \leq \gamma_s \leq 1$, which makes the UMPC a weighted combination of steady state (mean-level) control and a dynamic (relatively aggressive) controller tuned using the usual parameters, N_2, N_u etc. Two methods for the calculation of steady state prediction are presented. In order to stabilize plants with MPM, an option to use a conservative noise model for steady state prediction is provided

6.1 Introduction

LRPC's use a predicted future output trajectory or horizon $\{y_p(t+j), j = N_1, N_1+1, \dots N_2\}$ in the calculation of a set of optimal control moves $\{\Delta u(t+j-1)\ j=1, 2, \dots N_u\}$. Use of a larger output horizon, i.e. larger N_2, generally leads to more conservative, more robust control and hence is common in industrial applications. However, use of a larger N_2 also increases the computational and storage requirements of the control algorithm especially for adaptive controllers which require the solution of Diophantine equations at every time step. A relatively shorter horizon will reduce the computational load but it may give very vigorous control action and, in some cases, may destabilize the system. It is well-known that the step response coefficients of a stable process approach a steady state value. Therefore the later part of a step response can reasonably be replaced by a weighted steady state value This will result in substantial computational savings because several terms in the cost function are replaced by a single steady state prediction.

Steady state weighting was first introduced in LRPC's by Kwok (1992) and Kwok and Shah (1994), who included it in the minimization of the cost function of the GPC (Generalized Predictive Control). However their approach is complicated and has many limitations including the following:

- It is developed for a specific LRPC, i.e. GPC.
- It unnecessarily includes a set of steady state predictions instead of a single steady state prediction.
- The set of steady state predictions is treated in a matrix separate than the dynamic matrix making the formulation more complicated.
- The steady state weighting is not normalized, i.e. it runs from 0 to ∞.
- In most cases of Model Plant Mismatch (MPM) this approach to steady state weighting can not stabilize the system.

Appendix-6A summarizes the above mentioned GPC formulation with steady state error weighting due to Kwok and Shah.

In this chapter, single-point steady state output control is added to the original UMPC formulation so that the advantages of a long output horizon can be gained along with the computational savings associated with a smaller N_2 .

The proposed approach introduces a single, *normalized* tuning parameter $0 \leq \gamma_s \leq 1$, which permits the specification of any weighted combination of steady state (mean-level) control and the more aggressive control specified by the usual tuning parameters N_2 , N_u , etc. Tuning this single parameter , $0 \leq \gamma_s \leq 1$, provides a wide range of control responses and use of a "separated" Diophantine predictor in UMPC permits the use of multiple noise/disturbance models and/or a "disturbance horizon" which can be used to tune the dynamics (speed) of the output response to disturbances (independent of the servo response) and to compensate for the errors in the noise/disturbance model or prediction.

In summary, UMPC is a general, intuitive, efficient algorithm that includes *provision for independent tuning of the regulatory and servo responses.*

The present study proposes a much simpler, straightforward and robust formulation to incorporate the steady state weighting in LRPC's. The proposed formulation overcomes the above mentioned limitations of the existing formulation and includes the following features:

- It is based on a general LRPC rather than GPC
- It uses a new predictor which is much simpler than the classical predictor.
- It includes two methods to obtain the steady state prediction.
- It includes only a single steady state prediction rather than a set of steady state predictions and the rationale for including a single steady state prediction is mathematically established.
- The steady state prediction is a natural extension of the output horizon in the sense that the predicted steady state value is simply appended to the vector of predicted outputs.
- No additional matrices are required.
- No additional control law derivation is needed.
- The interpretation of the resulting control law is conceptually meaningful.
- The steady state weighting is normalized, i.e. it takes the values between 0 and 1.
- It allows the use of more conservative noise models for steady state prediction which is important for stabilizing systems with MPM.

6.2 The Generalized Measured Output Model and the Long Range Predictor

A slightly simplified form, corresponding to a single integrator, of the generalized measured output model introduced in Chapter-2 is:

$$y(t) = \frac{B}{A}u(t-1) + \frac{C}{D\breve{A}\Delta}e(t) \tag{6.1}$$

where $y(t)$, $u(t\text{-}1)$ and $e(t)$ are the output, the input and the uncertainty signals at time t respectively. A, B, C and D are polynomials in the backward shift operator q^{-1} and the notation $\breve{A}$ is used to indicate that the polynomial A shall only be included for equation error structures. More specifically:

$$\breve{A} = A \qquad \text{for EE structures} \tag{6.2}$$

$$\breve{A} = 1 \qquad \text{for OE structures} \tag{6.3}$$

The process transfer function is assumed to be stable since there is no concept of steady state for open-loop unstable processes.

The j-step ahead Separated Diophantine Predictor (SDP) is given as:

$$y_p(t+j|t) = \widetilde{P}_j \Delta u(t+j-1) + \overline{P}_j u^F(t-1) + F_j e^f(t) \tag{6.4}$$

where

$$\frac{C}{D\breve{A}\Delta} = E_j + q^{-j}\frac{F_j}{D\breve{A}\Delta} \tag{6.5}$$

$$\frac{B}{\Delta A} = \widetilde{P}_j + q^{-j}\frac{\overline{P}_j}{\Delta A} \tag{6.6}$$

and with the following definition of the filtered input and filtered error

$$u^F(t-1) = \frac{u(t-1)}{A} \qquad \text{and} \qquad e^f(t) = \frac{e(t)}{D\breve{A}\Delta} \tag{6.7}$$

Also

$$e^f(t) = \frac{e(t)}{D\breve{A}\Delta} = \frac{1}{C}\left[y(t) - \frac{B}{A}u(t-1)\right] = \frac{x(t)}{C} = x^f(t) \tag{6.8}$$

where

$$x(t) = \text{residue at } t = \left[y(t) - \frac{B}{A}u(t-1)\right] \tag{6.9}$$

$x^f(t)$ is the filtered residual at time t. Note that the residue $x(t)$ is independent of the noise model and therefore the filtered error $e^f(t)$ or the filtered residual $x^f(t)$ depends on the numerator of the noise model but is independent of its denominator. The separated Diophantine predictor (SDP) can be written in terms of the filtered residual as follows:

$$y_p(t+j|t) = \widetilde{P}_j \Delta u(t+j-1) + \overline{P}_j u^F(t-1) + F_j x^f(t) \tag{6.10}$$

SDP

The SDP has two parts. The first part consists of the first two terms and it exclusively depends on the plant model. The second part which is the last term in the SDP depends only on the employed noise model.

In most of the literature (e.g. Clarke, 1987a) on LRPC's the first j step response coefficients are represented by g_0 to g_{j-1} also $\widetilde{P}_j$ is denoted by $\widetilde{G}_j$. Therefore in the sequel $\widetilde{P}_j$ is replaced $\widetilde{G}_j$ which is defined as:

$$\widetilde{G}_j = g_0 + g_1 q^{-1} + g_2 q^{-2} + \cdots + g_{j-1} q^{-j+1} \tag{6.11}$$

The predictor in terms of filtered error signal is:

$$y_p(t+j|t) = \widetilde{G}_j \Delta u(t+j-1) + \overline{P}_j u^F(t-1) + F_j e^f(t) \tag{6.12}$$

and the predictor in terms of filtered residual is given as:

$$y_p(t+j|t) = \widetilde{G}_j \Delta u(t+j-1) + \overline{P}_j u^F(t-1) + F_j x^f(t) \tag{6.13}$$

or

$$y_p(t+j|t) = \widetilde{G}_j \Delta u(t+j-1) + f(t+j) \tag{6.14}$$

with

$$f(t+j) = \overline{P}_j u^F(t-1) + F_j x^f(t) \tag{6.15}$$

The first term in the predictor (6.13) (i.e. the forced response) can be rewritten in a summation form starting with the term involving the current control move (i.e. $\Delta u(t)$) to give the following predictor form:

$$y_p(t+j|t) = \sum_{i=1}^{j} g_{j-i}\Delta u(t+i-1) + \overline{P}_j u^F(t-1) + F_j x^f(t) \tag{6.16}$$

The above predictor is for the case where $N_u = j$ (N_u is the control horizon defined below). The following predictor is based on any $N_u \le j$.

$$y_p(t+j|t,N_u) = \sum_{i=1}^{N_u} g_{j-i}\Delta u(t+i-1) + \overline{P}_j u^F(t-1) + F_j x^f(t) \tag{6.17}$$

Note that an extra argument, N_u , has been specified in y_p nomenclature. This argument is often omitted when $N_u = j$.

The cost function in terms of the y_p nomenclature of (6.17) is given as:

$$J = \sum_{j=N_1}^{N_2} \gamma(j)\left[y_p(t+j|t,N_u) - w(t+j)\right]^2 + \sum_{j=1}^{N_u} \lambda(j)\left[\Delta u(t+j-1)\right]^2 \tag{6.18}$$

$$\mathbf{u} = \left[\mathbf{G}^{\mathrm{T}}\Gamma\mathbf{G} + \Lambda\right]^{-1}\mathbf{G}^{\mathrm{T}}\Gamma(\mathbf{w} - \mathbf{f}) \tag{6.19}$$

where

$\gamma(.)$ = the weighting on tracking error.

$y_p(.)$ = the predicted output values.

$w(.)$= the desired output values or the set-points.

$w_s(.)$= the desired output value or the set-point at steady state.

N_1 = the initial output horizon ≥ the earliest future output that is affected by the control move at time t.

N_2 = the final output horizon, the farthest future output that is included in the cost function J.

$\lambda(.)$ = the control weighting sequence.

$N_u =$ the control horizon, the number of future non-zero control moves.

$$\mathbf{u} = [\Delta u(t) \quad \Delta u(t+1) \quad \cdots \quad \Delta u(t+Nu-1)]^{\mathrm{T}} \tag{6.20}$$

$$\mathbf{w} = [w(t+N_1) \quad w(t+N_1+1) \quad \cdots \quad w(t+N_2)]^{\mathrm{T}} \tag{6.21}$$

$$\mathbf{f} = [f(t+N_1) \quad f(t+N_1+1) \quad \cdots \quad f(t+N_2)]^{\mathrm{T}} \tag{6.22}$$

$$\Gamma = \mathrm{diag}[\gamma(N_1) \quad \gamma(N_1+1) \quad \cdots \quad \gamma(N_2)] \tag{6.23}$$

$$\Gamma_s = \mathrm{diag}[\gamma_s(1) \quad \gamma_s(2) \quad \cdots \quad \gamma_s(N_u)] \tag{6.24}$$

$$\Lambda = \mathrm{diag}[\lambda(1) \quad \lambda(2) \quad \cdots \quad \lambda(N_u)] \tag{6.25}$$

$$\mathbf{G} = \begin{bmatrix} g_{N_1-1} & \cdots & \cdots & g_0 & \cdots & 0 \\ g_{N_1} & \cdots & g_2 & g_1 & \ddots & 0 \\ \vdots & \vdots & \vdots & & \cdots & g_0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ g_{N_2-1} & g_{N_2-2} & \cdots & \cdots & \cdots & g_{N_2-N_u} \end{bmatrix}_{N \times N_u} \tag{6.26}$$

6.3 Optimal Steady State Prediction

A brute force way to obtain the steady state prediction for a stable process is to use the above developed *j*-step ahead predictor (equation (6.10)) with large *j*. This would obviously require a lot of computational effort as it involves solution of Diophantine equations (6.5) and (6.6) to obtain $\widetilde{P}_j$ (or $\widetilde{G}_j$), $\overline{P}_j$ and F_j for $j \to \infty$ (i.e. for very large *j*). Note that in general only the last N_u coefficients of $\widetilde{G}_j$ are required to construct the dynamic matrix **G** corresponding to a control horizon of N_u. Moreover for steady state prediction based on $N_u = 1$ only one, the last, coefficient (represented by g_s) is needed.

Two simpler methods for steady state prediction calculation are proposed in the following sections.

6.3.1 The Steady State Prediction Method-I

The predictor (6.10) requires the steady state values of $\tilde{G}_j$, $\overline{P}_j$ and F_j for steady state output prediction. Setting $q^{-1} = 1$, to obtain steady state values, in the Diophantine equations (6.5) and (6.6) would give indeterminate results because of the presence of Δ which becomes zero at steady state.

In this method a Diophantine expansion for $\frac{B}{A\Delta}$ is derived in terms of the Diophantine equation for $\frac{B}{A}$, in order to get rid of Δ. The method utilizes the well-known fact (which can be easily proven) that for stable transfer functions the impulse response coefficients decay to zero with time.

Let

$$\frac{B}{A} = \tilde{H}_j + q^{-j}\frac{\overline{H}_j}{A} = h_0 + h_1 q^{-1} + h_2 q^{-2} + \cdots + h_{j-1} q^{-j+1} + q^{-j}\frac{\overline{H}_j}{A} \tag{6.27}$$

For stable A and as $j \to \infty$ both h_{j-1} and $\overline{H}_j$ diminish. Moreover

$$\left[\sum_{k=0}^{\infty} h_k\right] = \frac{B(1)}{A(1)} \qquad \text{for} \quad q^{-1} = 1 \quad \text{i.e. at steady state} \tag{6.28}$$

Now dividing (6.27) by Δ and noting that:

$$\frac{q^{-i}}{\Delta} = \sum_{k=i}^{j-1} q^{-k} + \frac{q^{-j}}{\Delta} \tag{6.29}$$

yields

$$\frac{B}{A\Delta} = h_0 \sum_{k=0}^{j-1} q^{-k} + h_1 \sum_{k=1}^{j-1} q^{-k} + h_2 \sum_{k=2}^{j-1} q^{-k} + \cdots + h_{j-1} \sum_{k=j-1}^{j-1} q^{-k} + \left[\sum_{k=0}^{j-1} h_k\right]\frac{q^{-j}}{\Delta} + q^{-j}\frac{\overline{H}_j}{A\Delta} \tag{6.30}$$

$$\begin{aligned} \frac{B}{A\Delta} = {} & \left[\sum_{k=0}^{0} h_k\right] + \left[\sum_{k=0}^{1} h_k\right] q^{-1} + \left[\sum_{k=0}^{2} h_k\right] q^{-2} \cdots + \left[\sum_{k=0}^{j-1} h_k\right] q^{-j+1} \\ & + \left[\sum_{k=0}^{j-1} h_k\right]\frac{q^{-j}}{\Delta} + q^{-j}\frac{\overline{H}_j}{A\Delta} \end{aligned} \tag{6.31}$$

$$\frac{B}{A\Delta}=\left[\sum_{k=0}^{0}h_k\right]+\left[\sum_{k=0}^{1}h_k\right]q^{-1}+\left[\sum_{k=0}^{2}h_k\right]q^{-2}\cdots+\left[\sum_{k=0}^{j-1}h_k\right]q^{-j+1}+q^{-j}\,\frac{\left[\sum_{k=0}^{j-1}h_k\right]A+\overline{H}_j}{A\Delta} \qquad (6.32)$$

Equation (6.32) is the Diophantine expansion for $\frac{B}{A\Delta}$ (equation (6.6) with $\widetilde{P}_j$ replaced by $\widetilde{G}_j$) with:

$$\widetilde{G}_j=\left[\sum_{k=0}^{0}h_k\right]+\left[\sum_{k=0}^{1}h_k\right]q^{-1}+\left[\sum_{k=0}^{2}h_k\right]q^{-2}\cdots+\left[\sum_{k=0}^{j-1}h_k\right]q^{-j+1} \qquad (6.33)$$

and

$$\overline{P}_j=\left[\sum_{k=0}^{j-1}h_k\right]A+\overline{H}_j \qquad (6.34)$$

For stable A and as $j\rightarrow\infty$ and using (6.28)

$$g_s = \text{ the last coefficient of } \widetilde{G}_j \text{ as } j\rightarrow\infty=\left[\sum_{k=0}^{\infty}h_k\right]=\frac{B(1)}{A(1)} \qquad (6.35)$$

and

$$\overline{P}_s=\left[\sum_{k=0}^{\infty}h_k\right]A=g_sA \qquad (6.36)$$

Similarly the following terms are obtained for the Diophantine equation (6.5) as $j\rightarrow\infty$ (the subscript s represents the steady state).

$$\begin{aligned} e_s &= \frac{C(1)}{D(1)\breve{A}(1)} \\ F_s &= e_sD\breve{A} \end{aligned} \qquad (6.37)$$

The following steady state output prediction for $N_u=1$ is obtained by substituting (6.35), (6.36) and (6.37) in equation (6.17):

$$y_p(s|t,N_u)=g_s\Delta u(t)+g_sAu^F(t-1)+e_sD\breve{A}x^f(t) \qquad (6.38)$$

6.3.2 The Steady State Prediction Method-II

This method is based on the fact that for stable transfer functions the step response coefficients and the residual converge to some constant value as time goes to infinity. This leads to the fact that at sufficiently large values of j two consecutive step response coefficients can be assumed to be identical.

The Diophantine expansions for a stable transfer function and sufficiently large j and j+1:

$$\begin{aligned} \frac{B}{A\Delta} &= g_0 + g_1 q^{-1} + g_2 q^{-2} + \cdots + g_j q^{-j+1} + q^{-j}\frac{\overline{P}_j}{A\Delta} \\ \frac{B}{A\Delta} &= g_0 + g_1 q^{-1} + g_2 q^{-2} + \cdots + g_j q^{-j+1} + g_j q^{-j} + q^{-j-1}\frac{\overline{P}_j}{A\Delta} \end{aligned} \tag{6.39}$$

subtracting the former equation from the later

$$\begin{aligned} 0 &= g_j q^{-j} + q^{-j-1}\frac{\overline{P}_j}{A\Delta} - q^{-j}\frac{\overline{P}_j}{A\Delta} = g_j - (1-q^{-1})\frac{\overline{P}_j}{A\Delta} = g_j - \frac{\overline{P}_j}{A} \\ \overline{P}_j &= g_j A \end{aligned} \tag{6.40}$$

As $j \rightarrow \infty$ the above relation becomes:

$$\overline{P}_s = g_s A \tag{6.41}$$

where the subscript s represents the steady state. Substituting this value of $\overline{P}_s$ in the first Diophantine equation gives

$$\begin{aligned} \frac{B}{A\Delta} &= g_0 + g_1 q^{-1} + g_2 q^{-2} + \cdots + g_s q^{-j+1} + q^{-j}\frac{g_s A}{A\Delta} \\ \frac{B}{A} &= \Delta g_0 + \Delta g_1 q^{-1} + \Delta g_2 q^{-2} + \cdots + \Delta g_s q^{-j+1} + q^{-j} g_s \end{aligned} \tag{6.42}$$

at steady state $q^{-1} = 1$, $\Delta = 0$ and $q^{-j} = 1$ therefore

$$g_s = \frac{B(1)}{A(1)} \tag{6.43}$$

Similar derivation yields e_s and F_s exactly as given in equation (6.37). The steady state output prediction is also the same as given in (6.38).

6.4 Pure Steady State Control with a Finite Control Horizon

The following steady state prediction for a finite control horizon N_u is obtained using (6.17) and the steady state values of various filters:

$$y_p(s|t,N_u)=\sum_{i=1}^{N_u}g_s\Delta u(t+i-1)+\overline{P}_s u^F(t-1)+F_s x^f(t) \tag{6.44}$$

where the s, as an argument or a subscript, represents the steady state. Thus $y_p(s|t,N_u)$ is the steady state prediction corresponding to control horizon N_u made at time t.

Kwok and Shah (1994) included a set of squares of the steady state tracking errors based on steady state predictions corresponding to control horizons 1 to N_u in the cost function to be minimized. They did not give any reason for it, moreover they did not even define explicitly the notion of the steady state prediction corresponding to a specific finite control horizon.

For N_u >1, an optimal control strategy based on the minimization of the square of the steady state tracking error, $y_p(s|t,N_u)-w(s)$, is an underdetermined problem (i.e. one equation with more than one variable). Note that $w(s)$ is the setpoint at steady state. In order to analyze the optimal pure steady state control consider the following set of steady state predictions:

$$\begin{array}{l}
y_p(s|t,1)=g_s\Delta u(t)+\overline{P}_s u^F(t-1)+F_s x^f(t)\\
y_p(s|t,2)=g_s\Delta u(t)+g_s\Delta u(t+1)+\overline{P}_s u^F(t-1)+F_s x^f(t)\\
\quad\vdots \qquad\quad \vdots \qquad\qquad \vdots \qquad\qquad \vdots \qquad\qquad \vdots\\
y_p(s|t,N_u)=\sum_{j=1}^{N_u}g_s\Delta u(t+j-1)+\overline{P}_s u^F(t-1)+F_s x^f(t)
\end{array} \tag{6.45}$$

and the cost function

$$J=\sum_{j=1}^{N_u}\gamma_s(j)\left[y_p(s|t,j)-w(s)\right]^2 \tag{6.46}$$

The control that minimizes J is:

$$\mathbf{u} = \left[\mathbf{G}_s^T \Gamma_s \mathbf{G}_s\right]^{-1} \mathbf{G}_s^T \Gamma_s (\mathbf{w}_s - \mathbf{f}_s) = \mathbf{G}_s^{-1}(\mathbf{w}_s - \mathbf{f}_s) \tag{6.47}$$

or in expanded form

$$\mathbf{u} = \begin{bmatrix} \Delta u(t) \\ \Delta u(t+1) \\ \vdots \\ \Delta u(t+N_u-1) \end{bmatrix} = \begin{bmatrix} g_s & 0 & \cdots & 0 \\ g_s & g_s & \ddots & \vdots \\ \vdots & & \ddots & 0 \\ g_s & \cdots & \cdots & g_s \end{bmatrix}^{-1} \begin{bmatrix} \varepsilon_s \\ \varepsilon_s \\ \vdots \\ \varepsilon_s \end{bmatrix} \tag{6.48}$$

Multiplication of (6.48) by $\mathbf{G}_s$ yields

$$\begin{bmatrix} g_s & 0 & \cdots & 0 \\ g_s & g_s & \ddots & \vdots \\ \vdots & & \ddots & 0 \\ g_s & \cdots & \cdots & g_s \end{bmatrix} \begin{bmatrix} \Delta u(t) \\ \Delta u(t+1) \\ \vdots \\ \Delta u(t+N_u-1) \end{bmatrix} = \begin{bmatrix} \varepsilon_s \\ \varepsilon_s \\ \vdots \\ \varepsilon_s \end{bmatrix} \tag{6.49}$$

and solving each of the equations in (6.49) gives

$$\begin{bmatrix} \Delta u(t) \\ \Delta u(t+1) \\ \vdots \\ \Delta u(t+N_u-1) \end{bmatrix} = \begin{bmatrix} \varepsilon_s / g_s \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{6.50}$$

Equation (6.50) clearly indicates that even if a set of N_u steady state predictions are considered, the optimal control action is to set all the control moves equal to zero except the first one. In other words there is no point in using a set of N_u steady state predictions corresponding to control horizons containing N_u points because *for steady state optimal control* (6.50) shows that only the first control move, $u(t)$, is non-zero. Moreover, as expected, the steady state control law corresponding to a control horizon of 1 is exactly identical to the mean-level control.

It follows that if steady state control is combined with control based on the minimization of the sum of squares of the dynamic tracking errors over a horizon from N_1 to N_2 the steady state contribution should be based on a control horizon of one regardless of the control horizon for the dynamic part. Otherwise (i.e. if the control horizon is greater than 1 for the steady state part) it can not strictly be called steady state control, and the combination can not rightly be termed as the LRPC with steady state weighting.

For pure steady state optimal control, the inclusion of the sum of squares of the steady state tracking errors corresponding to $N_u > 1$ as done by Kwok results in the control moves beyond time t being set equal to zero as shown in (6.50). However when the additional steady state tracking errors are used in combination with the dynamic tracking errors then the detuning effect of steady state weighting is not confined to the first control action. Instead it is spread over the control moves of the entire control horizon. Consequently, for a given steady state weighting, lesser detuning is observed as compared to the case when only a control horizon of one is used for the steady state prediction.

6.5 Combination of Steady State Control with LRPC

Incorporation of steady state weighting in the control law is straightforward. The basic control law undergoes only minor modifications. The steady state prediction is appended to the output horizon and a weighted square of the tracking error at steady state is added in the cost function as follows:

$$J = \sum_{j=N_1}^{N_2+1} \gamma(j)\left[y_p(t+j|t) - w(t+j)\right]^2 + \sum_{j=1}^{N_u} \lambda(j)\left[\Delta u(t+j-1)\right]^2$$

where (6.51)

$$y_p(t+N_2+1|t) = y_p(s|t,1) \quad \text{and} \quad \gamma(N_2+1) = \gamma(s)$$

The dynamic matrix is redefined as:

$$\mathbf{G} = \left[\begin{array}{cccccc} g_{N_1-1} & \cdots & \cdots & g_0 & \cdots & 0 \\ g_{N_1} & \cdots & g_2 & g_1 & \ddots & 0 \\ \vdots & \vdots & \vdots & & \cdots & g_0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ g_{N_2-1} & g_{N_2-2} & \cdots & \cdots & \cdots & g_{N_2-N_u} \\ \hdashline g_s & 0 & 0 & 0 & \cdots & 0 \end{array}\right]_{(N_2-N_1+2)\times N_u} \tag{6.52}$$

Similarly **f** and Γ are redefined to be:

$$\mathbf{f} = [f(t+N_1) \quad f(t+N_1+1) \quad \cdots \quad f(t+N_2) \mid f(s)]^T \tag{6.53}$$

where $f(s) =$ steady state value of free response

and

$$\Gamma = \mathrm{diag}[\gamma(N_1) \quad \gamma(N_1+1) \quad \cdots \quad \gamma(N_2) \mid \gamma(s)] \tag{6.54}$$

The above formulation offers a simple way of normalizing the weighting on the steady state control part. Normalization can be achieved by simply redefining the Γ as follows:

$$\Gamma = \mathrm{diag}[\gamma(N_1)[1-\gamma(s)] \quad \gamma(N_1+1)[1-\gamma(s)] \quad \cdots \quad \gamma(N_2)[1-\gamma(s)] \mid (N_2 - N_1 + 1)\gamma(s)] \tag{6.55}$$

The advantage of normalization is that the range of γ_s is now bounded between 0 to 1. Setting γ_s=0 means no steady state weighting at all, whereas γ_s=1 implies that the control is entirely based on the minimization of square of the steady state tracking error. Another obvious interpretation of the γ_s=1 case is that it is the mean-level control i.e. the control corresponding to an infinite output horizon and a control horizon of unity. Thus if the LRPC were designed so that γ_s=0 gave the most aggressive control that would ever be considered for the particular application (in the extreme, this would be $\lambda = 0$, $N_u = N_2$ for deadbeat control), then tuning the single parameter γ_s over the range 0 to 1 means that the optimal control action would vary from aggressive (deadbeat) to mean level. This is a significant simplification and provides the basis for an on-line "automatic" tuning mechanism.

6.6 Disturbance Modeling for Steady State Prediction

The roles of noise model in controller design include the following:

- It informs the controller about the expected dynamic structure of the noise and disturbances.
- An integrator in the noise model helps achieve an offset-free response.
- It can be used to increase/decrease the speed of disturbance rejection.

- It can stabilize plants with unmodeled dynamics.

Because of the stochastic nature of the noise and the disturbance, detailed identification of the noise model is practically unfeasible. In practice the noise model is specified as a tuning parameter to achieve the above mentioned goals.

An important role of the noise model is its influence on the speed of disturbance rejection. The primary objective in process control is regulatory control. In most applications the effect of the noise and disturbances must be compensated as quickly as possible. The polynomial D in the denominator of noise model (6.1) may be used to achieve the desired speed of disturbance rejection. DMC does not include such a polynomial in its noise model and therefore its speed of disturbance rejection is much lower than that of GPC which contains the process model denominator A in the denominator of its noise model. However the inclusion of A in the noise model is desirable only when the disturbances share the plant dynamics. In general, restricting the denominator polynomial to be A reduces the flexibility in specification of the noise model, and in many cases amplifies the effect of the unmodelled dynamics. Moreover the A in the noise model can not describe the measurement noise which is almost always added at the output i.e. it does not share the plant dynamics. A better way of controlling the speed of disturbance rejection is to have an arbitrary polynomial, D, in the denominator of the noise model rather than including the fixed process denominator polynomial A. The generalized noise model used in this study (6.1) was selected for this reason.

One simple way to achieve good speed of disturbance rejection without sacrificing stability is to control the contribution of the noise model to the future prediction. The extent of the effect of the noise model in the future prediction is termed the "*Disturbance Horizon*". In other words a performance-oriented noise model may be specified for some earlier part of the output horizon, then for the later part a more conservative noise model could be adopted. The term 'Disturbance Horizon" is suggested for this earlier part of the output horizon. The most logical choice of a sufficiently robust noise model is the DMC noise model. The use of two (or multiple in general) noise models in the suggested way over the output horizon results in the flexibility necessary to obtain an efficient robust controller.

One simple implementation of the disturbance horizon is as follows:

$$\begin{aligned} y_p(t+j|t) &= \widetilde{G}_j \Delta u(t+j-1) + \overline{P}_j u^F(t-1) + F_j x^f(t) && \text{for } j = N_1 \text{ to } N_d \\ y_p(t+j|t) &= \widetilde{G}_j \Delta u(t+j-1) + \overline{P}_j u^F(t-1) + x(t) && \text{for } j = N_{d+1} \text{ to } N_2 + 1 \end{aligned} \tag{6.56}$$

where N_d is the disturbance horizon. and $F_j = 1$ for DMC noise model.

Another option is to freeze the uncertainty contribution to prediction at the value corresponding to the end of the disturbance horizon as given in the following:

$$\begin{aligned} y_p(t+j|t) &= \widetilde{G}_j \Delta u(t+j-1) + \overline{P}_j u^F(t-1) + F_j x^f(t) && \text{for } j = N_1 \text{ to } N_d \\ y_p(t+j|t) &= \widetilde{G}_j \Delta u(t+j-1) + \overline{P}_j u^F(t-1) + F_{N_d} x(t) && \text{for } j = N_{d+1} \text{ to } N_2 + 1 \end{aligned} \tag{6.57}$$

The use of an ad hoc performance oriented noise model for the entire output horizon or for an earlier part of the output horizon is helpful in increasing the speed of disturbance rejection. However it is not suitable for the calculation of the steady state prediction in the presence of the unmodelled dynamics because such noise models amplify the effect of unmodelled dynamics and lead ultimately to instability. Consequently the steady state prediction should be based on the simplest possible noise model which happens to be the random walk model of DMC.

$$\frac{1}{\Delta} = \sum_{k=0}^{j-1} q^{-k} + \frac{q^{-j}}{\Delta} \tag{6.58}$$

or

$$F_j = 1 \quad \text{for any } j \text{, therefore } \quad F_s = 1 \tag{6.59}$$

Thus in calculating the free response at steady state $F_s = 1$ (or $F_s = F_{Nd}$) should be used in place of the one based on the prespecified performance oriented noise model. In other words the disturbance horizon should not extend beyond N_2 , it should be shorter than N_2 in case of severe MPM.

6.7 Simulation Examples

The role of steady state error weighting in LRPC, as developed in the preceding sections, is demonstrated in the following sub-sections through simulations using Matlab. The specific propositions set for the simulation examples were the following:

- The steady state error weighting, γ_s, detunes the closed-loop response.
- The detuning effect of γ_s is monotonic.
- The rate of detuning steeply decreases with γ_s .
- γ_s with a small N_2 gives the effect of a larger N_2 .
- Aggressive noise models destabilize or fail to stabilize systems with large MPM.
- In general γ_s weighting yields better performance than that obtained using λ-weighting.

Simulations are based on the following two processes (Process-A and Process-B) adopted from McIntosh (1988):

$$\text{Process - A (in } s \text{ - domain)} = \frac{1}{(1+s)(1+3s)(1+5s)} \tag{6.60}$$

which gives the following z-domain transfer function for a sampling interval of 1 second:

$$\text{Process - A (in z - domain)} = \frac{.00768z^{-1} + .02123z^{-2} + .00357z^{-3}}{1 - 1.9031z^{-1} + 1.1514z^{-2} - .2158z^{-3}} \tag{6.61}$$

The poles of Process-A in the are 0.8187, 0.7165 and 0.3679, while the zeros are at -2.586 and -0.1798. Note that one of the zeros is outside the unit circle making the discretized process nonminimum phase (NMP). The gain of the process is 1.

$$\text{Process - B (in } s \text{ - domain)} = \frac{2(229)}{(s+1)(s^2+30s+229)} \tag{6.62}$$

which gives the following z-domain transfer function for a sampling interval of 0.1 second:

$$\text{Process - B (in z - domain)} = \frac{.03700z^{-1} + .07172z^{-2} + .00785z^{-3}}{1 - 1.3422z^{-1} + .4455z^{-2} - .0450z^{-3}} \tag{6.63}$$

Process-B is the process used in Rohrs' (1984) benchmark example. The z-domain poles are 0.9048, 0.2187+.0443i and 0.2187-.0443i. Plant zeros are at -0.1164 and -1.822. Like Process-A one of the zeros is outside the unit circle making the discretized process nonminimum phase (NMP).

6.7.1 Example-1: Detuning Effect of γ_s-Weighting

This example demonstrates the detuning effect of γ_s . Figure-6.2 which is based on Process-A with no MPM shows the effect of low values of γ_s (0 to 0.2). The higher values of γ_s (0.3 to 0.9) are used in Figure 6.3. It can be seen that the detuning rate is much higher in the lower range of γ_s . Figures 6.4 and 6.5 show a similar γ_s effect for Process-B with no unmodelled dynamics. All simulations in this example are based on $N_1 = 1$, $N_2 = 5$, $N_u = 1$ and $\lambda = 0$.

6.7.2 Example-2: γ_s-Weighting to mimic a larger N_2

The original motivation for including γ_s-weighting into the MPC algorithm was to mimic the effect of a large output horizon while using a much shorter one. This point is illustrated in Figures 6.6 to 6.9. Figures 6.6 and 6.7 show that a γ_s value of 0.3 with $N_2 = 5$ gives a response almost identical to $N_2 = 20$ and $\gamma_s = 0$. Figure 6.6 is based on a DMC noise model whereas a GPC noise model is used for the simulation shown in Figure 6.7. Obviously the GPC noise model results in much more aggressive disturbance rejection than the DMC model. Both simulations are based on Process-A with no MPM and $N_1 = 1$, $N_u = 1$ and $\lambda = 0$. Figure 6.8 demonstrates the N_2 -reducing effect of γ_s-weighting in presence of severe MPM. In this simulation the third order Process-A is used with the following first order model

McIntosh (1988) has indicated that the above model has severe mismatch and the GPC based on this model is very difficult to stabilize. As shown in Figure 6.8, a γ_s value of 0.35 with an N_2 of as low as 10 gives results comparable to $N_2 = 120$ with $\gamma_s = 0$. The simulation is based on the DMC noise model. All simulations used Process-A with $N_1 = 1$, $N_u = 1$ and $\lambda = 0$.

6.7.3 Example-3: Use of DMC noise model for steady state prediction

The noise model has a profound effect on the steady state prediction especially in the presence of MPM. In most cases of unmodelled dynamics the controlled system based on a GPC noise model (no T-filter, i.e. $C = 1$) can not be stabilized using γ_s-weighting. This is because of the fact that in steady state prediction the GPC noise model amplifies the current, at time = t, MPM (or the unmodeled dynamics) by order magnitudes. This fact is depicted in Figure 3.2 where the steady state uncertainty contribution to the output prediction (for a frequency $\omega = \pi/2$) is about 100 times larger than its value at time = t. A DMC-based controller is stabilizable by γ_s-weighting but gives undesirably slow disturbance rejection. In order to obtain stable control with good disturbance rejection properties the controller is based on two noise models. The GPC noise model is used for the entire output horizon except the steady state prediction which is calculated on the basis of the DMC noise model. Figure 6.9 clearly illustrates the advantage of using two noise models. The simulations in this figure use Process-B with the following reduced order model

$$\text{Reduced order model for Process - B } = \frac{0.145z^{-1}}{1-0.93z^{-1}} \tag{6.64}$$

The model has substantial MPM. As can be seen, a GPC based controller exhibits unstable behavior while a DMC noise model gives very slow disturbance rejection. However, a combination of GPC and DMC yields stable control with much faster disturbance rejection performance. All simulations in the figure use $N_1 = 1$, $N_2 = 10$, $N_u = 1$, $\gamma_s = 0.1$ and $\lambda = 0$.

6.7.4 Example-4: Comparison between λ and γ_s-Weightings

Use of the control weighting (λ-weighting) in LRPC tuning is well established. This example demonstrates that γ_s-weighting with DMC noise modeling yields much better results than λ-weighting. Figure 6.10 compares the two strategies for Process-B with following first order model:

$$\text{Reduced order model for Process - B } = \frac{0.033z^{-1}}{1-0.94z^{-1}} \tag{6.65}$$

The λ-weighting strategy uses λ=2 and the γ_s-weighting employs $\gamma_s = 0.02$. It is clear that γ_s-weighting gives better results. The simulations in this figure use $N_1 = 1$, $N_2 = 10$, and N_u =2. Similar results are shown in Figure 6.11 where simulations are carried out for $N_1 = 1$, $N_2 = 10$, N_u =2 and Process-A with the following reduced order model:

$$\text{Reduced order model for Process - A } = \frac{0.08z^{-1}}{1-0.94z^{-1}} \tag{6.66}$$

Again tuning with γ_s-weighting is superior to λ-weighting.

6.8 Conclusions

A completely new formulation for incorporating the steady state weighting in model predictive control is developed. The γ_s-weighting in the MPC algorithm provides a new tuning methodology that is no more difficult to implement than increasing N_2 by one. The effect of a large output horizon is obtained by using γ_s-weighting on the predicted steady state error with a much shorter output horizon. The new formulation is based on a generalized noise/disturbance model and uses a new separated Diophantine predictor which is much simpler than the classical lumped Diophantine predictor. Two methods to obtain the steady state prediction are presented. The new formulation uses only a single steady state prediction rather than a set of steady state predictions as required in the existing methodology (Kwok, 1992). The steady state-only control is analyzed and the use of a single steady state tracking error in the cost function is mathematically justified. The steady state prediction is implemented as a natural extension of the output horizon so that no extra matrices or vectors are involved and the original form of the control law is preserved. The resulting control law is interpreted as an interpolation between regular MPC and mean level MPC. The steady state weighting is normalized (equation (6.55)), i.e. it takes the values between 0 and 1. The steady state prediction is optionally based on conservative noise models, e.g. DMC, to stabilize systems with model plant mismatch (MPM). Simulation examples are presented to illustrate the advantages of γ_s-weighting.

References:

Clarke, D.W., Mohtadi, C. and Tuffs, P.S., " Generalized Predictive Control - Part I the Basic Algorithm ", Automatica, Vol. 23, No. 2, pp.137-148, 1987a.

Clarke, D.W., Mohtadi, C. and Tuffs, P.S., " Generalized Predictive Control - Part II Extensions and Interpretations ", Automatica, Vol. 23, No. 2, pp. 149-160, 1987b.

Clarke, D.W.," Adaptive Generalized Predictive Control ", Proceedings of the Fourth International Conference on Process Control (CPCIV), Editors Y. Arkun and W. H. Ray, Padre Island, TX, 1991.

Cutler, C.R. and Ramaker, B.L., "Dynamic Matrix Control - A Computer Control Algorithm ", AIChE National Meeting, Houston, TX, 1979.

Kwok, K., "Long Range Adaptive Predictive Control ", Ph.D. thesis, University of Alberta, 1992.

Kwok, K., and Shah, S. L., "Long Range Predictive Control with a Terminal Matching Condition", Chemical Engineering Science, Vol. 49, No. 9, pp. 1287-1300, 1994.

McIntosh, A.R., "Performance and Tuning of Adaptive Generalized Predictive Control ", M.Sc. thesis, University of Alberta, 1988.

Rohrs, C.E., Athans, M., Valavani, L. and Stein, G., "Some Design Guidelines for Discrete-time Adaptive Controllers", Automatica, Vol. 20, No. 5, pp. 653-660, 1984.

Appendix 6-A

Review of the Existing Formulation for GPC with Steady State Error Weighting

Following is a summary of the Kwok et. al.(1994) formulation for the incorporation of the steady state error in GPC.

The ARIMAX output model used in GPC is:

$$y(t) = \frac{B}{A}u(t-d) + \frac{C}{A\Delta}e(t) \tag{6.67}$$

The optimal *j*-step ahead predictor is given as:

$$y_p(t+j|t) = \tilde{G}_j \Delta u(t+j-d) + \overline{G}_j \frac{\Delta u(t-d)}{C} + F_j \frac{y(t)}{C} \tag{6.68}$$

The optimal *steady state* predictor is derived as:

$$y_p(\mathrm{s}|t) = g_s \sum_{j=1}^{Nu} \Delta u(t+j-d) + \overline{G}_s \frac{\Delta u(t-d)}{C} + F_s(q^{-1}) \frac{y(t)}{C} \tag{6.69}$$

The following two Diophantine equations are required as in GPC.

$$\frac{C}{A\Delta} = E_j + q^{-j} \frac{F_j}{A\Delta} \tag{6.70}$$

$$\frac{E_j B}{C} = \tilde{G}_j + q^{-j} \frac{\overline{G}_j}{C} \tag{6.71}$$

The cost function is modified by adding a summation of steady state error terms as follows:

$$\begin{aligned} J = {} & \sum_{j=N_1}^{N_2} \gamma(j)\left[y_p(t+j|t) - w(t+j)\right]^2 + \sum_{j=1}^{N_u} \lambda(j)[\Delta u(t+j-1)]^2 \\ & + \sum_{j=1}^{N_u} \gamma_s(j)\left[y_p(s|t+j-1) - w(s)\right]^2 \end{aligned} \tag{6.72}$$

The control law that minimizes (6.72) is :

$$\mathbf{u} = \left[\mathbf{G}^{\mathrm{T}}\Gamma\mathbf{G} + \Lambda + \underline{\mathbf{G}_{s}^{\mathrm{T}}\Gamma\mathbf{G}_{s}}\ \right]^{-1}\left[\mathbf{G}^{\mathrm{T}}\Gamma(\mathbf{w} - \mathbf{f}) + \underline{\mathbf{G}_{s}^{\mathrm{T}}\Gamma_{s}(\mathbf{w}_{s} - \mathbf{f}_{s})}\ \right] \tag{6.73}$$

where

$\gamma(.)$ = the weighting on tracking error.

$\gamma_s(.)$ = the weighting on tracking error at steady state.

$y_p(.)$ = the predicted output values.

$w(.)$= the desired output values or the set-points.

$w_s(.)$= the desired output value or the set-point at steady state.

N_1 = the initial output horizon ≥ the earliest future output that is affected by the control move at time t.

N_2 = the final output horizon, the farthest future output that is included in the cost function J.

$\lambda(.)$ = the control weighting sequence.

N_u = the control horizon, the number of future non-zero control moves.

$$\mathbf{u} = [\Delta u(t) \quad \Delta u(t+1) \quad \cdots \quad \Delta u(t + Nu - 1)]^{\mathrm{T}} \tag{6.74}$$

$$\mathbf{w} = [w(t+N_1) \quad w(t+N_1+1) \quad \cdots \quad w(t+N_2)]^{\mathrm{T}} \tag{6.75}$$

$$\begin{aligned} &\mathbf{f} = [f(t+N_1) \quad f(t+N_1+1) \quad \cdots \quad f(t+N_2)]^{\mathrm{T}} \\ &\text{where} \quad f(t+j) = \overline{G}_j \frac{\Delta u(t-d)}{C} + F_j \frac{y(t)}{C} \end{aligned} \tag{6.76}$$

$$\mathbf{w}_{s}(N_u \times N_u) = [1 \quad 1 \quad \cdots \quad 1]^{\mathrm{T}} w(s) \tag{6.77}$$

$$\mathbf{f}_{s} = [1 \quad 1 \quad \cdots \quad 1]^{\mathrm{T}} . \overline{G}_s \frac{\Delta u(t-d)}{C} + F_s \frac{y(t)}{C} \tag{6.78}$$

$$\Gamma = \mathrm{diag}[\gamma(N_1) \quad \gamma(N_1+1) \quad \cdots \quad \gamma(N_2)] \tag{6.79}$$

$$\Gamma_{s} = \mathrm{diag}[\gamma_s(1) \quad \gamma_s(2) \quad \cdots \quad \gamma_s(N_u)] \tag{6.80}$$

$$\Lambda = \mathrm{diag}[\lambda(1) \quad \lambda(2) \quad \cdots \quad \lambda(N_u)] \tag{6.81}$$

$$\mathbf{G} = \begin{bmatrix} g_{N_1-1} & g_{N_1-2} & \cdots & g_0 & 0 & \cdots & 0 \\ g_{N_1} & g_{N_1-1} & \cdots & g_1 & g_0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & & \ddots & g_0 \\ \vdots & \vdots & \ddots & \ddots & & & g_1 \\ \vdots & \vdots & & \ddots & & & \vdots \\ g_{N_2-1} & g_{N_2-2} & \cdots & \cdots & \cdots & \cdots & g_{N_2-N_u} \end{bmatrix} \tag{6.82}$$

$$\mathbf{G_s} = \begin{bmatrix} g_s & 0 & \cdots & 0 \\ g_s & g_s & \ddots & \vdots \\ \vdots & & \ddots & 0 \\ g_s & \cdots & \cdots & g_s \end{bmatrix}_{N_u \times N_u} \tag{6.83}$$

$$g_s = \frac{B(1)}{A(1)}\ ; \qquad e_s = \frac{C(1)}{A(1)}; \quad F_s = e_s A \quad \text{and} \quad \overline{G}_s \Delta = g_s C - e_s B$$

The main difference between this implementation of steady state weighting and the one developed in this thesis for UMPC is the last summation term in (6.72), the underlined steady state terms in (6.73), the use of separated Diophantine predictor (SDP) and the calculation of the free responses, $\mathbf{f}_s$.

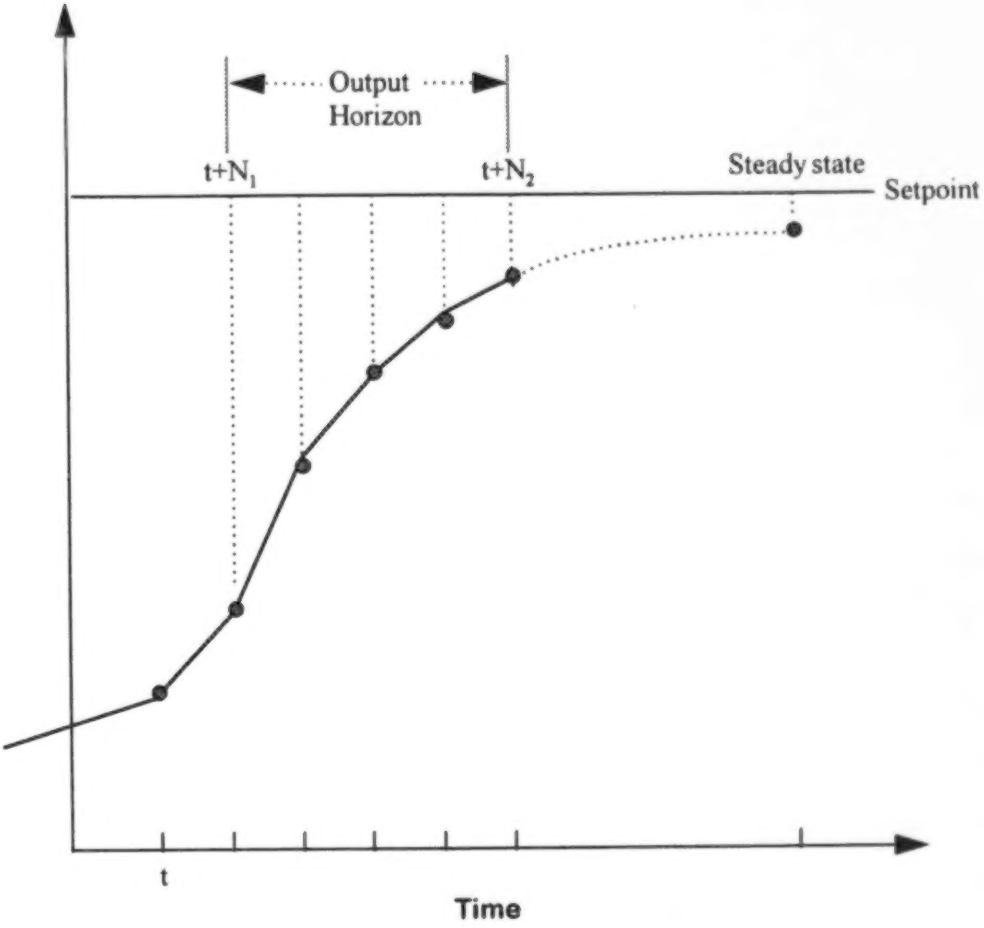

Output
Horizon
t+N1
t+N2
Steady state
Setpoint
t
Time

Figure 6.1 Incorporation of steady state error in long-range predictive control

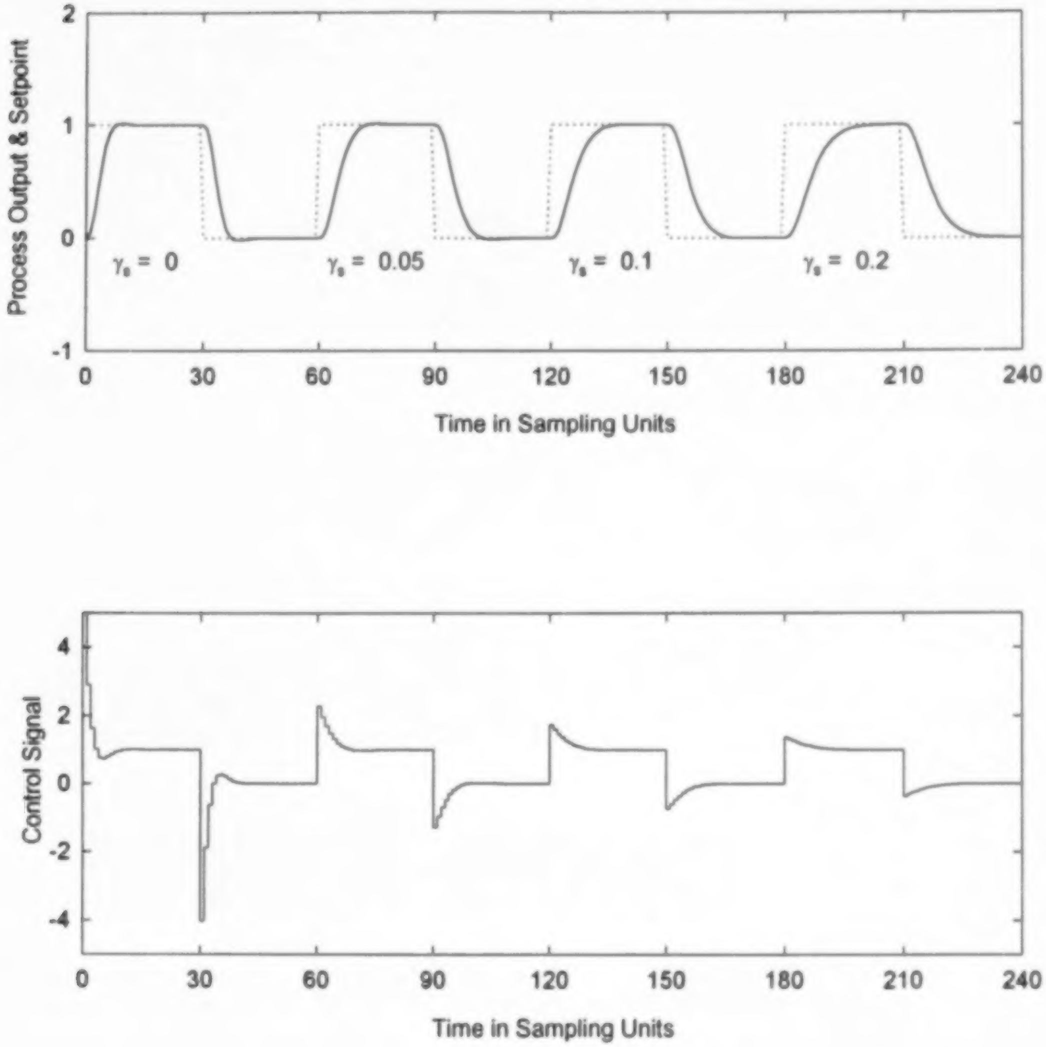


Figure 6.2 Detuning with steady state error weighting for process A, γ_s=0 to 0.2

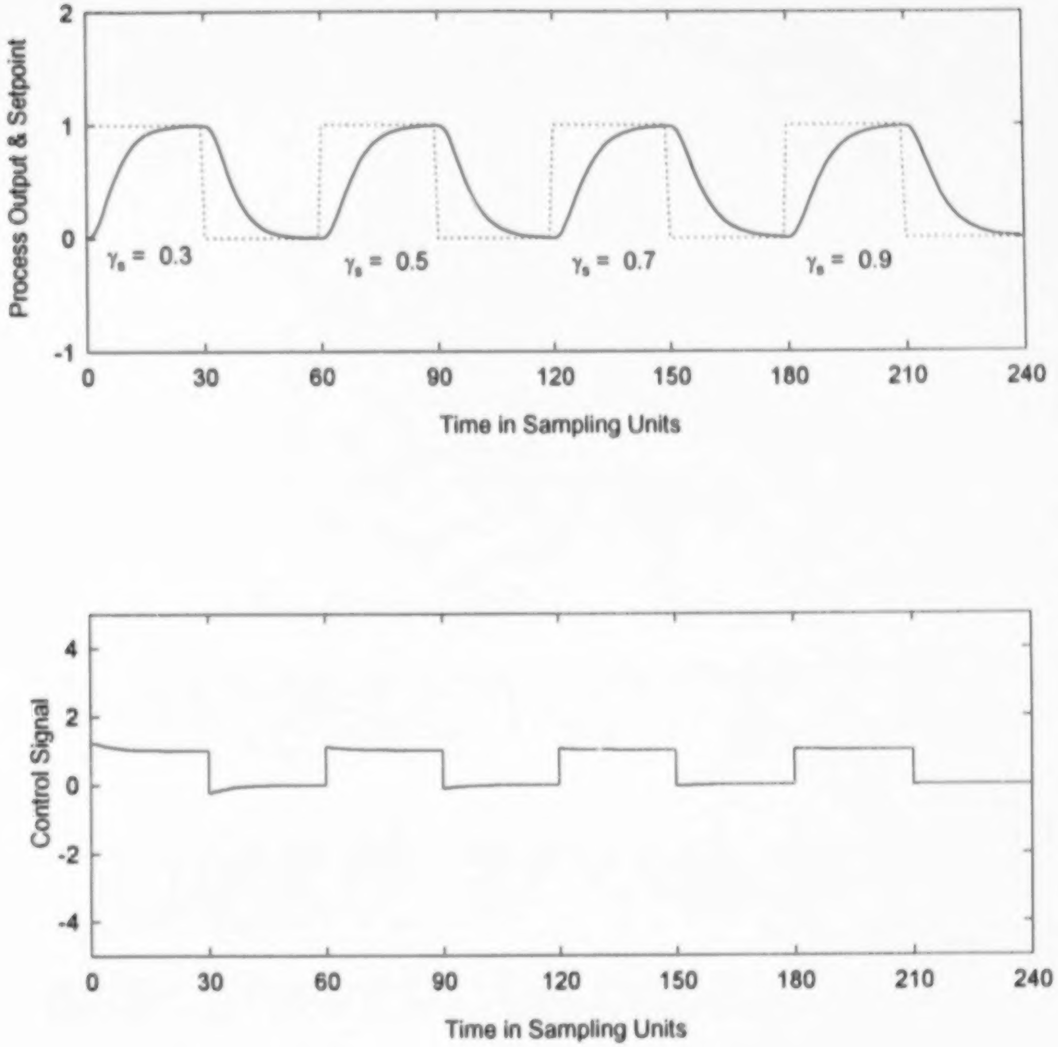


Figure 6.3 Detuning with steady state error weighting for process A, γ_s=0.3 to 0.9

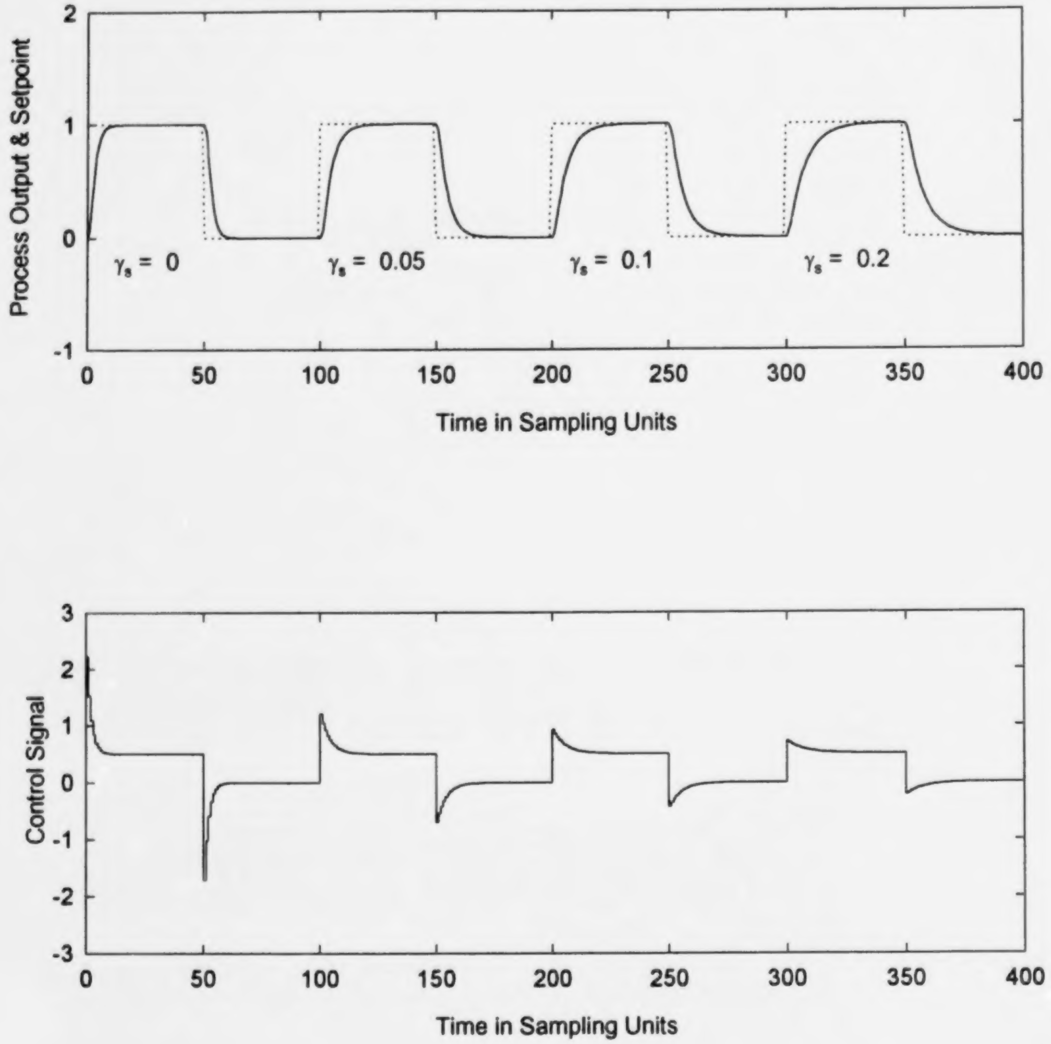


Figure 6.4 Detuning with steady state error weighting for process B, γ_s=0 to 0.2

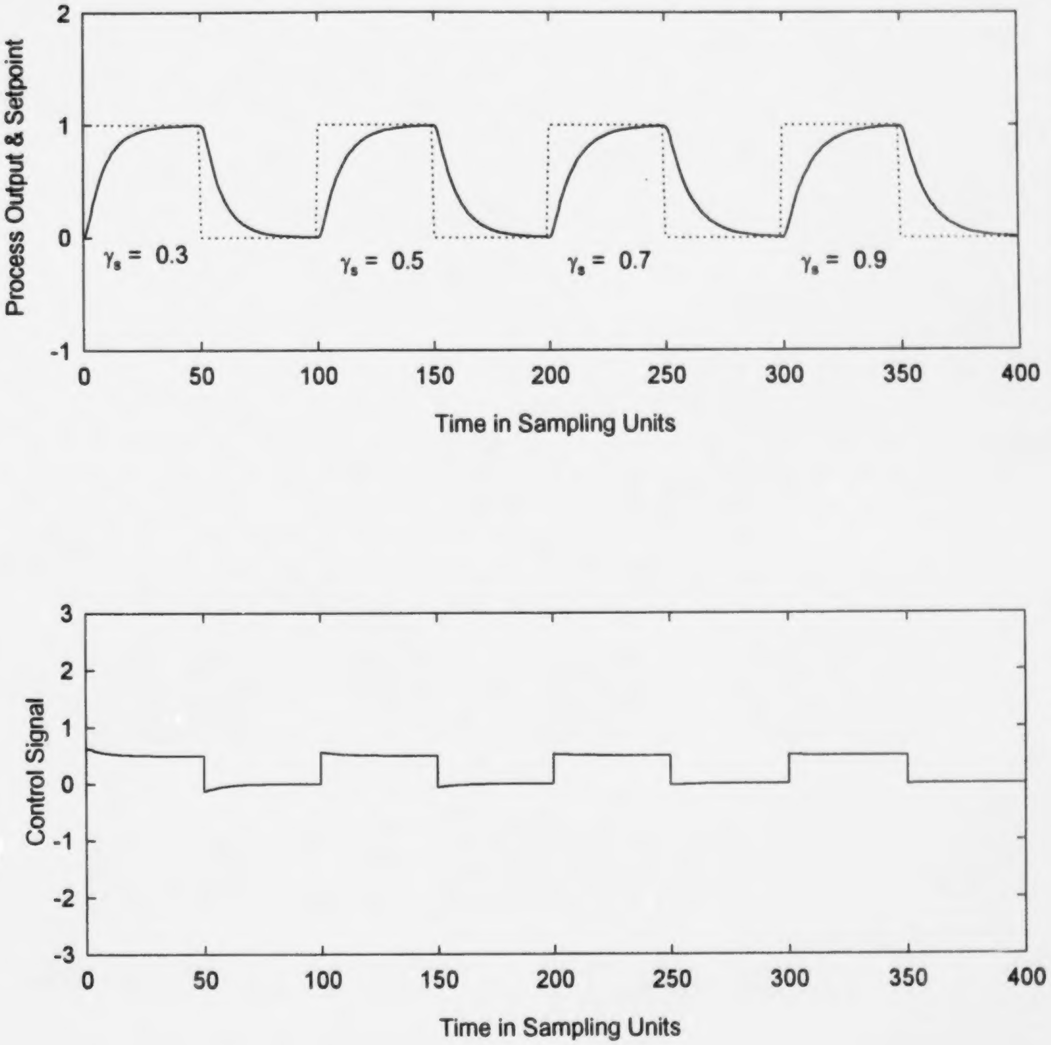


Figure 6.5 Detuning with steady state error weighting for process B, γ_s=0.3 to 0.9

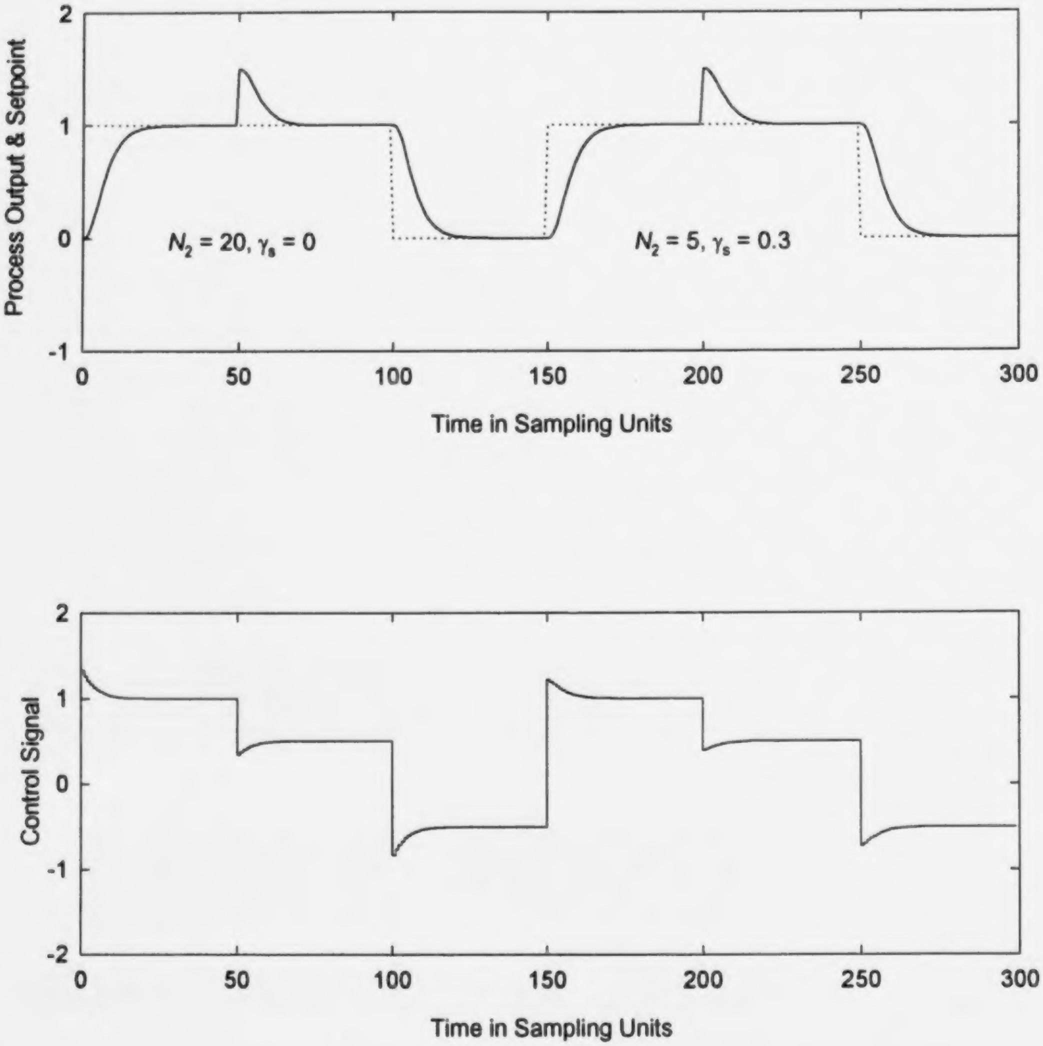


Figure 6.6 A small N_2 plus γ_s gives an effect of a larger N_2, DMC for Process-A

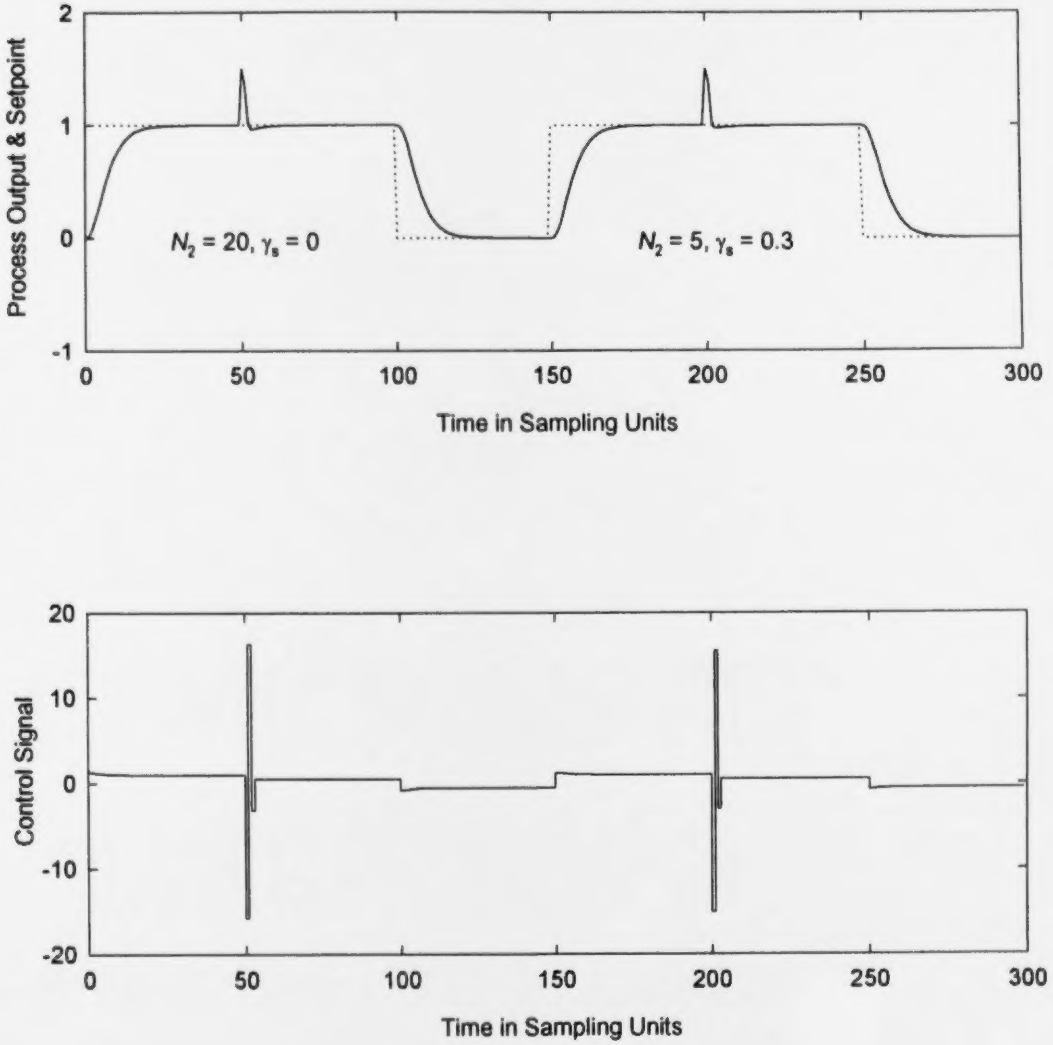


Figure 6.7 A small N_2 plus γ_s gives an effect of a larger N_2, GPC for Process-A

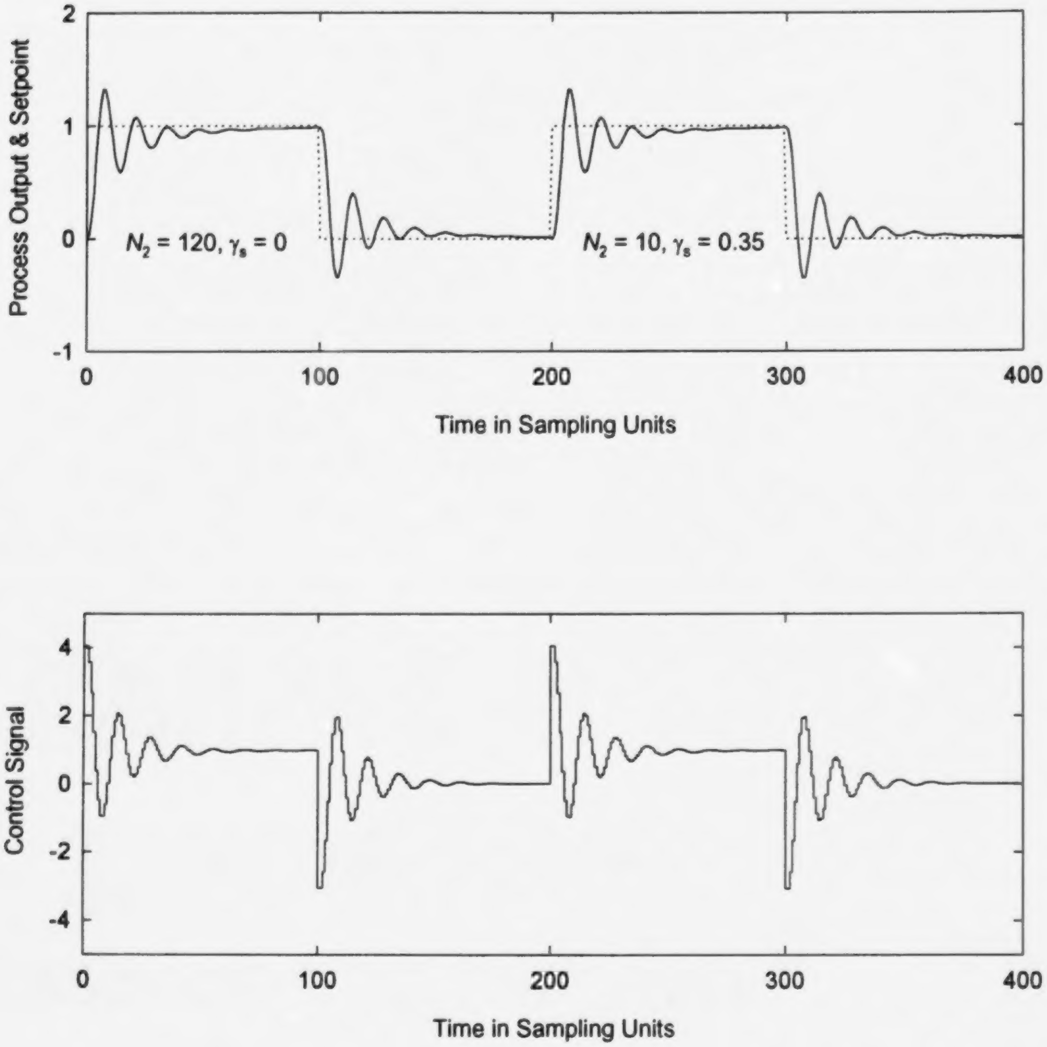


Figure 6.8 A small N_2 plus γ_s gives an effect of a larger N_2, DMC for Process-A with severe Model Plant Mismatch

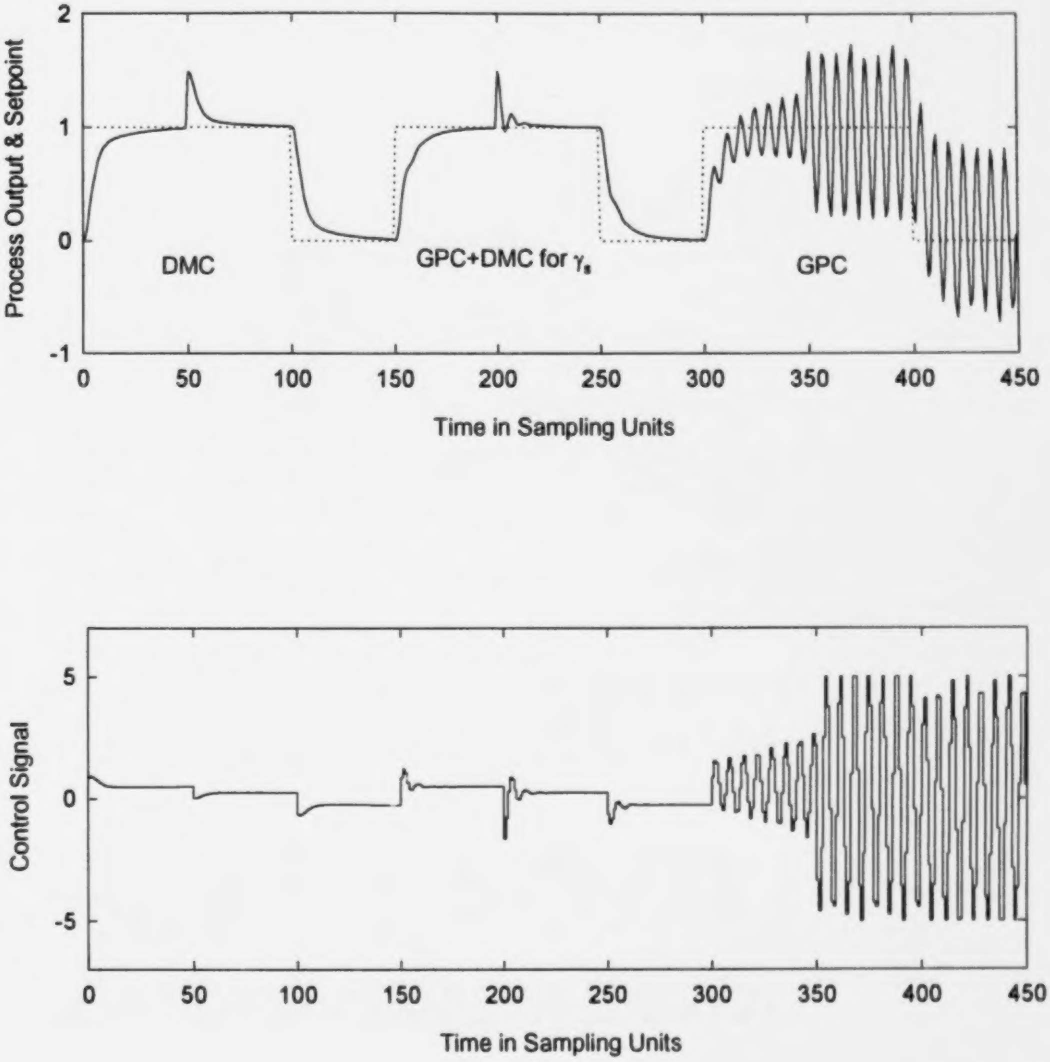


Figure 6.9 Effect of noise model structure for steady state prediction calculation

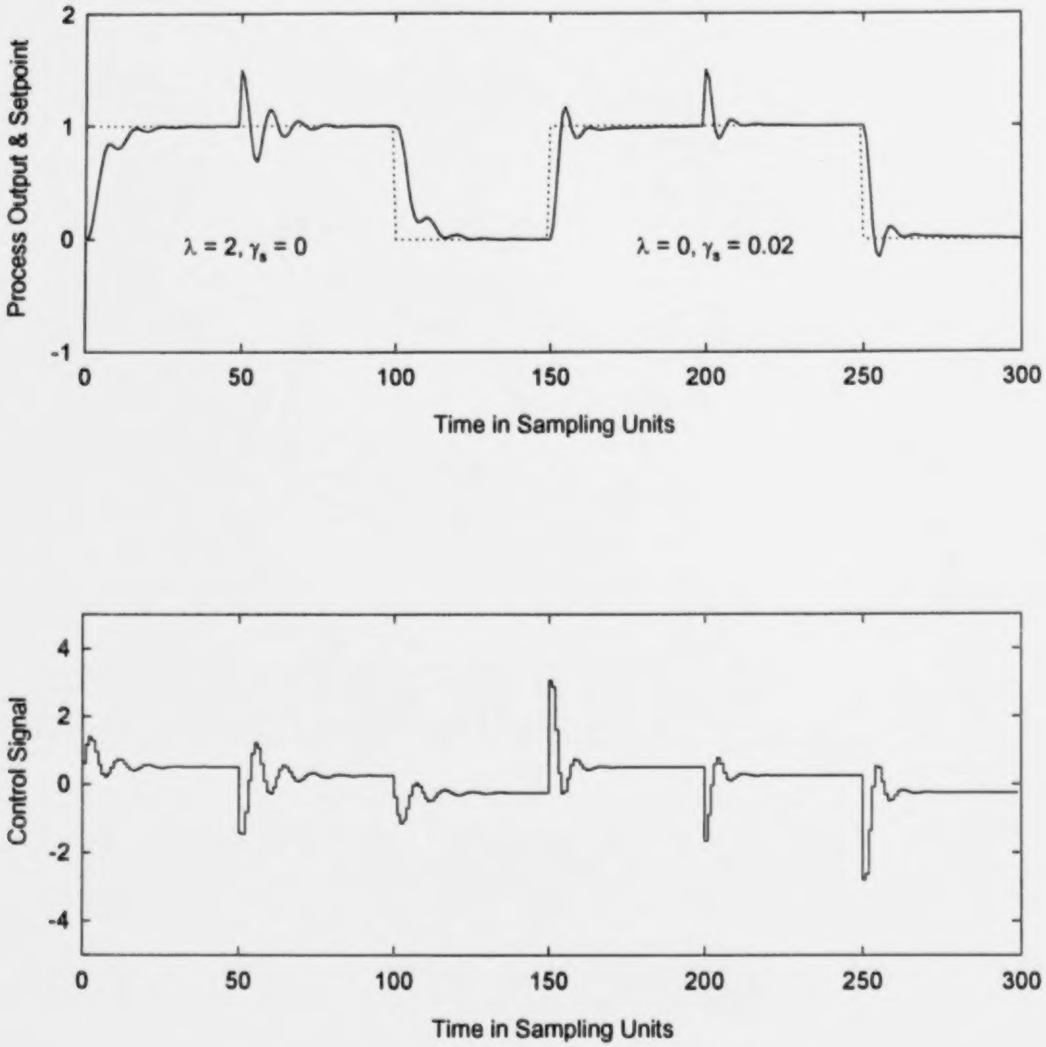


Figure 6.10 γ_s-weighting *vs* λ-weighting, GPC for Process-B with MPM

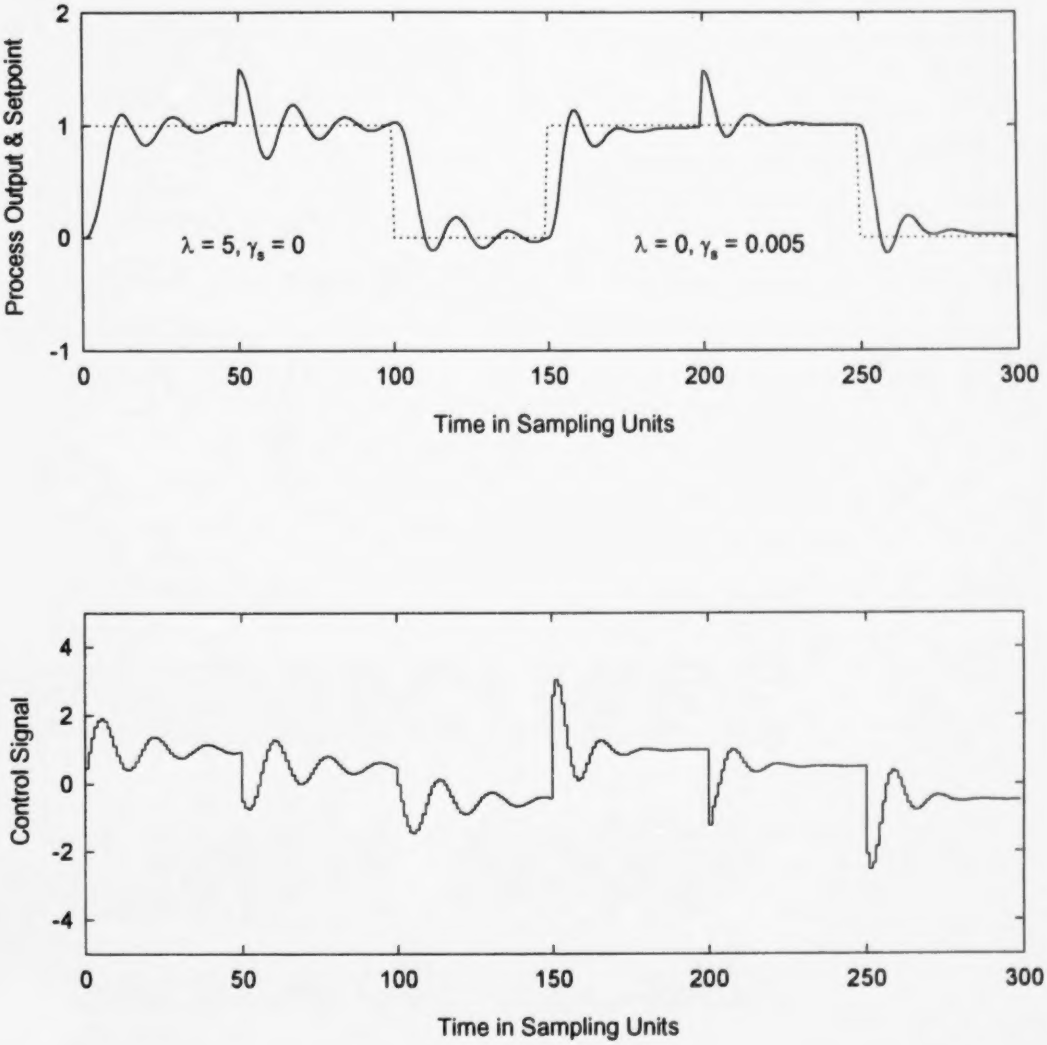


Figure 6.11 γ_s-weighting *vs* λ-weighting, GPC for Process-A with MPM

Chapter 7

Interpolation and Recursion of Control Horizon in Long Range Predictive Control

Long Range Predictive Control (LRPC) typically involves the calculation of N_u discrete future control moves that minimize the control (tracking) error over an output horizon of $N_2 > N_u$ points. The control horizon is an important design/tuning variable but, by definition, has integer values. This chapter shows how the equivalent of non-integer (fractional) values of N_u can be obtained by "interpolating" between the control produced by two integer values of N_u. One simple method is to stack the blocks of weighted errors prior to optimization. This method involves higher dimension matrices. The second method of interpolation uses the algebra of partitioning of matrices to reduce the dimension of the involved matrices. In both methods the interval of interpolation, m, may be more than unity.

A recursive formulation based on matrix partitioning is presented that permits the control action corresponding to N_u+m to be calculated recursively starting from the control moves for N_u. The recursive formulation is quite general in that the recursion may be done for more than one step (the m-step ahead recursion).

Finally an interpolative recursive formulation is developed. This method combines the interpolation and the recursion of the control horizon into one algorithm. Again both the recursion and the interpolation can be multi-step.

7.1 Introduction

Unified Model Predictive Control (UMPC) falls into the general classification of LRPC. The main feature of an LRPC is that it uses a predicted future output trajectory (called the output horizon) of the system to calculate a set of optimal control moves. The control horizon is defined as the number of independent, non-zero control moves permitted in the optimization. The control horizon is inherently integer. However in many cases, especially in the presence of model plant mismatch (MPM), an interpolation between two integer levels of the control horizon is highly desirable for tuning purposes.

Matrix inversion is an important implementation issue for control horizons beyond 3. A recursive formulation is therefore quite attractive from a practical point of view since it can decrease the dimensions of the matrices to be inverted.

Both of the above mentioned formulations are developed in the following sections.

7.2 Review of Conventional Long Range Predictive Control Algorithm

To date most of the popular long-range predictive controllers use the following type of quadratic cost functions for the optimal control calculation (Cutler, 1979, Clarke, 1987a; De Keyser, 1991):

$$J = \sum_{j=1}^{N} \gamma(j)\left[y_p(t + N_1 + j - 1|t) - w(t + j)\right]^2 + \sum_{j=1}^{N_u} \lambda(j)[\Delta u(t + j - 1)]^2 \qquad (7.1)$$

where the various variables have usual meanings (cf. Chapter-2)
Note that for the above cost function:

$$N = N_2 - N_1 + 1 \qquad (7.2)$$

The prediction $y_p(t+j|t)$ in the above cost function is usually based on a stochastic model for the measured output such as an ARMAX or ARIMAX model. In UMPC the following Box-Jenkin's type model is employed:

$$y(t) = \frac{B}{A}u(t-1) + \frac{C}{D\breve{A}\Delta}e(t) \tag{7.3}$$

where $y(t)$, $u(t\text{-}1)$ and $e(t)$ are the output, the input and the uncertainty signals at time t respectively. A, B, C and D are polynomials in the backward shift operator q^{-1} and the notation $\breve{A}$ is used to indicate that the polynomial A shall only be included for equation error structures. More specifically:

$$\breve{A} = A \qquad \text{for EE structures} \tag{7.4}$$

$$\breve{A} = 1 \qquad \text{for OE structures} \tag{7.5}$$

The above model is quite general and controllers like GPC and DMC are special cases. GPC is obtained by setting $\breve{A} = A$, and $D = 1$. The setting $\breve{A} = 1$, and $C = D = 1$ gives the DMC model structure (Clarke, 1991). The following j-step ahead separated Diophantine predictor (SDP) developed in Chapter 3 is based on the generalized output model of equation (7.3):

$$y_p(t+j|t) = \tilde{G}_j\Delta^n u(t+j-1) + \overline{P}_j u^F(t-1) + F_j x^f(t) \tag{7.6}$$

$$\tilde{G}_j = g_0 + g_1 q^{-1} + \ldots + g_{j-1}q^{-j+1} \tag{7.7}$$

The various filters and terms of (7.6) are defined in Chapter 3.

The prediction, $y_p(t+j|t)$, consists of two parts. The first term on the RHS of the predictor (7.6) is the response due to the future input moves. This first part is known as the forced response. The sum of the last two terms on the RHS of the predictor is called the free-response. It is the response due to the past inputs $u(.)$ and disturbances up to time t.

The predictor (7.6) can also be given as:

$$y_p(t+j|t) = \tilde{G}_j \Delta^n u(t+j-1) + f(t+j) \tag{7.8}$$

where $f(t+j)$ is the free response at time = $t+j$.

The first term in the predictor (7.6) (i.e. forced response) can be rewritten in a summation form starting with the term involving the current control move (i.e. $\Delta u(t)$) to give the following predictor form:

$$y_p(t+j|t) = \sum_{i=1}^{j} g_{j-i} \Delta u(t+i-1) + \overline{P}_j u^F(t-1) + F_j x^f(t) \tag{7.9}$$

The above predictor is for the case where $N_u = j$ (note that N_u is the control horizon). The following predictor is based on any $N_u \leq j$.

$$y_p(t+j|t, N_u) = \sum_{i=1}^{N_u} g_{j-i} \Delta u(t+i-1) + \overline{P}_j u^F(t-1) + F_j x^f(t) \tag{7.10}$$

Note that an extra argument, N_u, has been specified for y_p nomenclature. This argument is usually not included when $N_u = j$ as in equations (7.6) and (7.9).

The control law for the minimization of (7.1) is given as (McIntosh, 1988):

$$\mathbf{u} = \left[\mathbf{G}^{\mathrm{T}} \Gamma \mathbf{G} + \Lambda\right]^{-1} \mathbf{G}^{\mathrm{T}} \Gamma (\mathbf{w} - \mathbf{f}) \tag{7.11}$$

where

$$\mathbf{u} = [\Delta u(t) \quad \Delta u(t+1) \quad \cdots \quad \Delta u(t+Nu-1)]^{\mathrm{T}} \tag{7.12}$$

$$\mathbf{w} = [w(t+N_1) \quad w(t+N_1+1) \quad \cdots \quad w(t+N_2)]^{\mathrm{T}} \tag{7.13}$$

$$\mathbf{f} = [f(t+N_1) \quad f(t+N_1+1) \quad \cdots \quad f(t+N_2)]^{\mathrm{T}} \tag{7.14}$$

$$\Gamma = \mathrm{diag}[\gamma(N_1) \quad \gamma(N_1+1) \quad \cdots \quad \gamma(N_2)] \tag{7.15}$$

$$\Lambda = \mathrm{diag}[\lambda(1) \quad \lambda(2) \quad \cdots \quad \lambda(N_u)] \tag{7.16}$$

$$\mathbf{G}=\begin{bmatrix} g_{N_1-1} & \cdots & \cdots & g_0 & \cdots & 0 \\ g_{N_1} & \cdots & g_2 & g_1 & \ddots & 0 \\ \vdots & \vdots & \vdots & & \cdots & g_0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ g_{N_2-1} & g_{N_2-2} & \cdots & \cdots & \cdots & g_{N_2-N_u} \end{bmatrix}_{N \times N_u} \tag{7.17}$$

7.3 Control Horizon Interpolation Method-I

The LRPC formulation as developed above is good only for the integer values of the *control horizon*, N_u. In many cases a low value of N_u yields very slow control while some higher control horizon, say N_u+m, gives control that is too active. Usually the higher control horizon is employed and some other detuning parameter, e.g. the control weighting, λ, is used to stabilize the response. A weighted combination of the control action generated for N_u and N_u+m would provide a very direct method of tuning and could be interpreted as an interpolation of the control horizon between N_u and N_u+m.

The above described interpolation is obtained by minimizing a weighted sum of tracking errors corresponding to two control horizons, N_u+m and N_u with weightings μ and $1-\mu$ respectively, instead of a single control horizon N_u. The cost function to achieve this "interpolation" is:

$$\begin{aligned} J = \sum_{j=1}^{N}(1-\mu)\gamma(j)\left[y_p(t+N_1+j-1|t,N_u)-w(t+j)\right]^2 + \sum_{j=1}^{N_u}\lambda(j)\left[\Delta u(t+j-1)\right]^2 \\ + \sum_{j=1}^{N}\mu\gamma(j)\left[y_p(t+N_1+j-1|t,N_u+m)-w(t+j)\right]^2 \end{aligned} \tag{7.18}$$

Note that the extra argument, N_u, in y_p has been used as defined in (7.10) in order to identify the control law corresponding to each control horizon.

A simple way to incorporate the two control horizons N_u, and N_u+m with weightings $1-\mu$ and μ respectively using the cost function (7.18) is to define $\mathbf{G}^a$, Γ^a, $\mathbf{w}^a$, $\mathbf{f}^a$ and Λ^a as follows:

$$\mathbf{G}^* = \begin{bmatrix} \mathbf{G} & \mathbf{G}_m \\ \mathbf{G} & 0 \end{bmatrix} \qquad \Gamma^* = \begin{bmatrix} \mu\Gamma & 0 \\ 0 & (1-\mu)\Gamma \end{bmatrix} \qquad \mathbf{w}^* = \begin{bmatrix} \mathbf{w} \\ \mathbf{w} \end{bmatrix} \qquad \mathbf{f}^* = \begin{bmatrix} \mathbf{f} \\ \mathbf{f} \end{bmatrix} \tag{7.19}$$
$$\Lambda^* = \begin{bmatrix} \Lambda & 0 \\ 0 & \Lambda_m \end{bmatrix}$$

where **G**, **w**, **f** and Γ are the same as defined in section 7.2 and $\mathbf{G}_m$ and Λ_m are defined as:

$$\mathbf{G}_m = \begin{bmatrix} 0 & \cdots & \cdots \\ 0 & \cdots & \cdots \\ g_0 & \cdots & \vdots \\ \vdots & \vdots & \vdots \\ g_{N2-Nu-1} & \cdots & g_{N2-Nu-m} \end{bmatrix}_{N \times m} \tag{7.20}$$

and

$$\Lambda_m = \mathrm{diag}[\lambda(N_u+1) \quad \lambda(N_u+2) \quad \cdots \quad \lambda(N_u+\mathrm{m})] \tag{7.21}$$

The equivalent of equation (7.11) is then:

$$\mathbf{u} = (\mathbf{G}^{*T}\Gamma^*\mathbf{G}^* + \Lambda^*)^{-1}\mathbf{G}^*\Gamma^*(\mathbf{w}^* - \mathbf{f}^*) \tag{7.22}$$

This method is simple, however it doubles the row size of most of the matrices. Note that the setpoints and the free responses are exactly the same for the two control horizons.

7.4 Control Horizon Interpolation Method-II

This section proposes an improvement of the above method for control horizon interpolation. Unlike the method given in section 7.3, the proposed method does not increase the dimensions of the various matrices. The main idea is to expand different matrices in Method-I to get lower dimensional matrices.

$$\mathbf{G}^* = \begin{bmatrix} \mathbf{G}_{+m} \\ \mathbf{G}_{+0} \end{bmatrix} \qquad \mathbf{G}_{+m} = [\mathbf{G} \quad \mathbf{G}_m] \qquad \mathbf{G}_{+0} = [\mathbf{G} \quad 0] \tag{7.23}$$

$$\mathbf{G}^{*T}\Gamma^*\mathbf{G}^* = \begin{bmatrix} \mathbf{G}_{+m}^T & \mathbf{G}_{+0}^T \end{bmatrix} \begin{bmatrix} \mu\Gamma & 0 \\ 0 & (1-\mu)\Gamma \end{bmatrix} \begin{bmatrix} \mathbf{G}_{+m} \\ \mathbf{G}_{+0} \end{bmatrix} = \mathbf{G}_{+m}^T \mu\Gamma\mathbf{G}_{+m} + \mathbf{G}_{+0}^T(1-\mu)\Gamma\mathbf{G}_{+0} \tag{7.24}$$

$$\mathbf{G}^{\#T}\Gamma^{\#}(\mathbf{w}^{\#}-\mathbf{f}^{\#})=\begin{bmatrix}\mathbf{G}_{+m}^{T} & \mathbf{G}_{+0}^{T}\end{bmatrix}\begin{bmatrix}\mu\Gamma & 0\\ 0 & (1-\mu)\Gamma\end{bmatrix}\begin{bmatrix}\mathbf{w}-\mathbf{f}\\ \mathbf{w}-\mathbf{f}\end{bmatrix}$$

$$=\mathbf{G}_{+m}^{T}\mu\Gamma(\mathbf{w}-\mathbf{f})+\mathbf{G}_{+0}^{T}(1-\mu)\Gamma(\mathbf{w}-\mathbf{f}) \qquad (7.25)$$

$$=\left[\mathbf{G}_{+m}^{T}\mu\Gamma+\mathbf{G}_{+0}^{T}(1-\mu)\Gamma\right](\mathbf{w}-\mathbf{f})$$

Therefore the control law (7.11) becomes

$$\mathbf{u}=\left[\mathbf{G}_{+m}^{T}\mu\Gamma\mathbf{G}_{+m}+\mathbf{G}_{+0}^{T}(1-\mu)\Gamma\mathbf{G}_{+0}+\Lambda^{\#}\right]^{-1}\left[\mathbf{G}_{+m}^{T}\mu\Gamma+\mathbf{G}_{+0}^{T}(1-\mu)\Gamma\right](\mathbf{w}-\mathbf{f}) \qquad (7.26)$$

For $\mu = 0$ the above equation reduces to the original control equation for the control horizon N_u provided the trailing zero columns are deleted from $\mathbf{G}_{+0}$. For $\mu = 1$ it gives the original control equation for the control horizon N_u+m.

7.5 Control Horizon Recursion

The use of a larger control horizon requires inversion of higher dimensional matrices. A recursive solution would be both cheaper in terms of computational load, and more robust in terms of numerical properties. The following development for a control horizon-recursive solution of MPC algorithm is based on the well-known matrix inversion lemma.

7.5.1 Recursive Formulation

Define $\mathbf{G}_{+m}$ as follows:

$$\mathbf{G}_{+m}=[\mathbf{G} \quad \mathbf{G}_{m}] \qquad (7.27)$$

where $\mathbf{G}$ and $\mathbf{G}_{m}$ are the same as defined in Section 7.3. Note that $\mathbf{G}_{+m}$ is the dynamic matrix corresponding to a control horizon of N_u+m. The control law for this latter case (using equation (7.11)) is:

$$\mathbf{u}_{+m}=\left[\mathbf{G}_{+m}^{T}\Gamma\mathbf{G}_{+m}+\Lambda^{\#}\right]^{-1}\mathbf{G}_{+m}^{T}\Gamma(\mathbf{w}-\mathbf{f}) \qquad (7.28)$$

where

$\mathbf{u}_{+\mathrm{m}} = \begin{bmatrix} \mathbf{u}^{\#} \\ \mathbf{u}^{\#}_{\mathrm{m}} \end{bmatrix}$ $\mathbf{u}^{\#}$ and $\mathbf{u}^{\#}_{\mathrm{m}}$ are vectors of sizes N_u and m respectively

The various terms in (7.28) are evaluated as follows:

$$\mathbf{G}^{\mathrm{T}}_{+\mathrm{m}}\Gamma\mathbf{G}_{+\mathrm{m}} + \Lambda^{\#} = \begin{bmatrix} \mathbf{G}^{\mathrm{T}} \\ \mathbf{G}^{\mathrm{T}}_{\mathrm{m}} \end{bmatrix}\Gamma[\mathbf{G} \quad \mathbf{G}_{\mathrm{m}}] + \begin{bmatrix} \Lambda & 0 \\ 0 & \Lambda_{\mathrm{m}} \end{bmatrix} = \begin{bmatrix} \mathbf{G}^{\mathrm{T}}\Gamma\mathbf{G} + \Lambda & \mathbf{G}^{\mathrm{T}}\Gamma\mathbf{G}_{\mathrm{m}} \\ \mathbf{G}^{\mathrm{T}}_{\mathrm{m}}\Gamma\mathbf{G} & \mathbf{G}^{\mathrm{T}}_{\mathrm{m}}\Gamma\mathbf{G}_{\mathrm{m}} + \Lambda_{\mathrm{m}} \end{bmatrix} \tag{7.29}$$

$$= \begin{bmatrix} \mathbf{X} & \mathbf{Z} \\ \mathbf{Z}^{\mathrm{T}} & \mathbf{S} \end{bmatrix}$$

where

$$\mathbf{X} = \mathbf{G}^{\mathrm{T}}\Gamma\mathbf{G} + \Lambda, \quad \mathbf{Z} = \mathrm{G}^{\mathrm{T}}\Gamma\mathrm{G}_{\mathrm{m}}, \quad \mathbf{Z}^{\mathrm{T}} = \mathbf{G}^{\mathrm{T}}_{\mathrm{m}}\Gamma\mathbf{G}, \quad \text{and} \quad \mathbf{S} = \mathbf{G}^{\mathrm{T}}_{\mathrm{m}}\Gamma\mathbf{G}_{\mathrm{m}} + \Lambda_{\mathrm{m}} \tag{7.30}$$

Applying the matrix inversion lemma

$$\left[\mathbf{G}^{\mathrm{T}}_{+\mathrm{m}}\Gamma\mathbf{G}_{+\mathrm{m}} + \Lambda^{\#}\right]^{-1} = \begin{bmatrix} \mathbf{X} & \mathbf{Z} \\ \mathbf{Z}^{\mathrm{T}} & \mathbf{S} \end{bmatrix}^{-1} = \begin{bmatrix} \mathbf{X}^{-1} - \mathbf{W}\mathbf{Z}^{\mathrm{T}}\mathbf{X}^{-1} & \mathbf{W} \\ \mathbf{W}^{\mathrm{T}} & \mathbf{V} \end{bmatrix} \tag{7.31}$$

where

$$\mathbf{W} = -\mathbf{X}^{-1}\mathbf{Z}\mathbf{V} \quad ,\mathbf{W}^{\mathrm{T}} = -\mathbf{V}^{\mathrm{T}}\mathbf{Z}^{\mathrm{T}}\mathbf{X}^{-1} \quad \text{and} \quad \mathbf{V} = (\mathbf{S} - \mathbf{Z}^{\mathrm{T}}\mathbf{X}^{-1}\mathbf{Z})^{-1} \tag{7.32}$$

The dimensions of the various matrices are as follows:

$$\begin{array}{llllll} \mathbf{G} \text{ is } N \times N_u & \mathbf{G}_{\mathrm{m}} \text{ is } N \times m & \mathbf{G}_{+\mathrm{m}} \text{ is } N \times (N_u + m) \\ \mathbf{G} \text{ is } N \times N & \Lambda^{\#} \text{ is } (N_u + m) \times (N_u + m) & \\ \Lambda \text{ is } N_u \times N_u & \Lambda_m \text{ is } m \times m & \mathbf{X} \text{ is } N_u \times N_u \\ \mathbf{Z} \text{ is } N_u \times m & \mathbf{S} \text{ is } m \times m & \mathbf{V} \text{ is } m \times m \\ \mathbf{W} \text{ is } N_u \times m & & \end{array} \tag{7.33}$$

Postmultiplication of (7.31) by $\mathbf{G}^{\mathrm{T}}_{+\mathrm{m}}\Gamma(\mathbf{w} - \mathbf{f})$ gives

$$\left[\mathbf{G}^{\mathrm{T}}_{+\mathrm{m}}\Gamma\mathbf{G}_{+\mathrm{m}} + \Lambda^{\#}\right]^{-1}\mathbf{G}^{\mathrm{T}}_{+\mathrm{m}}\Gamma(\mathbf{w} - \mathbf{f}) =$$

$$= \begin{bmatrix} \mathbf{X} & \mathbf{Z} \\ \mathbf{Z}^{\mathrm{T}} & \mathbf{S} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{G}^{\mathrm{T}} \\ \mathbf{G}^{\mathrm{T}}_{\mathrm{m}} \end{bmatrix}\Gamma = \begin{bmatrix} \mathbf{X}^{-1} - \mathbf{W}\mathbf{Z}^{\mathrm{T}}\mathbf{X}^{-1} & \mathbf{W} \\ \mathbf{W}^{\mathrm{T}} & \mathbf{V} \end{bmatrix} \begin{bmatrix} \mathbf{G}^{\mathrm{T}} \\ \mathbf{G}^{\mathrm{T}}_{\mathrm{m}} \end{bmatrix}\Gamma$$

$$= \begin{bmatrix} (\mathbf{X}^{-1} - \mathbf{W}\mathbf{Z}^{\mathrm{T}}\mathbf{X}^{-1})\mathbf{G}^{\mathrm{T}}\Gamma + \mathbf{W}\mathbf{G}^{\mathrm{T}}_{\mathrm{m}}\Gamma \\ \mathbf{W}^{\mathrm{T}}\mathbf{G}^{\mathrm{T}}\Gamma + \mathbf{V}\mathbf{G}^{\mathrm{T}}_{\mathrm{m}}\Gamma \end{bmatrix}$$

(7.34)

Using (7.28) $\mathbf{u}_{+m} = \begin{bmatrix} \mathbf{u}^{\#} \\ \mathbf{u}_m^{\#} \end{bmatrix}$ is obtained as:

$$\begin{bmatrix} \mathbf{u}^{\#} \\ \mathbf{u}_m^{\#} \end{bmatrix} = \begin{bmatrix} \mathbf{X} & \mathbf{Z} \\ \mathbf{Z}^T & \mathbf{S} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{G}^T \\ \mathbf{G}_m^T \end{bmatrix} \Gamma(\mathbf{w}-\mathbf{f}) = \begin{bmatrix} (\mathbf{X}^{-1} - \mathbf{W}\mathbf{Z}^T\mathbf{X}^{-1})\mathbf{G}^T\Gamma + \mathbf{W}\mathbf{G}_m^T\Gamma \\ \mathbf{W}^T\mathbf{G}^T\Gamma + \mathbf{V}\mathbf{G}_m^T\Gamma \end{bmatrix}(\mathbf{w}-\mathbf{f})$$

$$= \begin{bmatrix} (\mathbf{I} - \mathbf{W}\mathbf{Z}^T)\mathbf{X}^{-1}\mathbf{G}^T\Gamma(\mathbf{w}-\mathbf{f}) + \mathbf{W}\mathbf{G}_m^T\Gamma(\mathbf{w}-\mathbf{f}) \\ \mathbf{W}^T\mathbf{G}^T\Gamma(\mathbf{w}-\mathbf{f}) + \mathbf{V}\mathbf{G}_m^T\Gamma(\mathbf{w}-\mathbf{f}) \end{bmatrix} \tag{7.35}$$

or in terms of **u** (the control moves vector corresponding to the control horizon of N_u)

$$\begin{bmatrix} \mathbf{u}^{\#} \\ \mathbf{u}_m^{\#} \end{bmatrix} = \begin{bmatrix} (\mathbf{I} - \mathbf{W}\mathbf{Z}^T)\mathbf{u} + \mathbf{W}\mathbf{G}_m^T\Gamma(\mathbf{w}-\mathbf{f}) \\ \mathbf{W}^T\mathbf{G}^T\Gamma(\mathbf{w}-\mathbf{f}) + \mathbf{V}\mathbf{G}_m^T\Gamma(\mathbf{w}-\mathbf{f}) \end{bmatrix}$$

$$= \begin{bmatrix} (\mathbf{I} + \mathbf{X}^{-1}\mathbf{Z}\mathbf{V}\mathbf{Z}^T)\mathbf{u} - \mathbf{X}^{-1}\mathbf{Z}\mathbf{V}\mathbf{G}_m^T\Gamma(\mathbf{w}-\mathbf{f}) \\ -\mathbf{V}^T\mathbf{Z}^T\mathbf{u} + \mathbf{V}\mathbf{G}_m^T\Gamma(\mathbf{w}-\mathbf{f}) \end{bmatrix} \tag{7.36}$$

$$\begin{aligned} \mathbf{u}^{\#} &= (\mathbf{I} + \mathbf{X}^{-1}\mathbf{Z}\mathbf{V}\mathbf{Z}^T)\mathbf{u} - \mathbf{X}^{-1}\mathbf{Z}\mathbf{V}\mathbf{G}_m^T\Gamma(\mathbf{w}-\mathbf{f}) \\ &= \mathbf{u} + \mathbf{X}^{-1}\mathbf{Z}\mathbf{V}\left[\mathbf{Z}^T\mathbf{u} - \mathbf{G}_m^T\Gamma(\mathbf{w}-\mathbf{f})\right] \end{aligned} \tag{7.37}$$

$$\mathbf{u}_m^{\#} = -\mathbf{V}^T\mathbf{Z}^T\mathbf{u} + \mathbf{V}\mathbf{G}_m^T\Gamma(\mathbf{w}-\mathbf{f}) = -\mathbf{V}^T\left[\mathbf{Z}^T\mathbf{u} - \mathbf{V}\mathbf{G}_m^T\Gamma(\mathbf{w}-\mathbf{f})\right] \tag{7.38}$$

7.5.2 Recursive Implementation

The recursive formulation developed in section 7.5.1 is quite general in that it allows for recursion of control horizon by *m* steps. However it requires an *m*×m matrix inversion and calculation of base **u** (i.e. **u** corresponding to control horizons N_u). A recursive implementation, for a control horizon of N_u, without matrix inversion can be developed by setting *m*=1 and starting with N_u =1. Moreover the recursion of **X** (i.e. (7.31)) rather than **u** (i.e. (7.36)) is more efficient. The following procedure is adopted:

Set $\mathbf{G}_1$ = the first column of **G** (**G** is the original dynamic matrix)

$$\mathbf{X}_1 = \mathbf{G}_1^{\mathrm{T}}\Gamma\mathbf{G}_1 + \Lambda \text{ a scalar and } \mathbf{X}_1^{-1} = \frac{1}{\mathbf{X}_1}$$

1. Start with $k = 1$
2. Set $\mathbf{G}_k$ = the first k columns of $\mathbf{G}$
3. Define $\mathbf{g}_{k+1}$ = the $k+1$ th column of $\mathbf{G}$
4. $\mathbf{X}_{k+1}^{-1} = \begin{bmatrix} \mathbf{X}_k^{-1} - \mathbf{W}_k\mathbf{Z}_k^{\mathrm{T}}\mathbf{X}_k^{-1} & \mathbf{W}_k \\ \mathbf{W}_k^{\mathrm{T}} & \mathbf{V}_k \end{bmatrix}$
5. Set $k = k + 1$ if $k < N_u$ and go to step 2 or quit if $k = N_u$

where

$$\mathbf{Z}_k = \mathbf{G}_k^{\mathrm{T}}\Gamma\mathbf{g}_{k+1}\ ,\ \mathbf{S}_k = \mathbf{g}_{k+1}^{\mathrm{T}}\Gamma\mathbf{g}_{k+1} + \lambda_{k+1}\ ,\mathbf{V}_k = \left[\mathbf{S}_k - \mathbf{Z}_k^{\mathrm{T}}\mathbf{X}_k^{-1}\mathbf{Z}_k\right]^{-1} \text{ and } \mathbf{W}_k = -\mathbf{X}_k^{-1}\mathbf{Z}_k\mathbf{V}_k$$

Note that $\mathbf{S}_k - \mathbf{Z}_k^{\mathrm{T}}\mathbf{X}_k^{-1}\mathbf{Z}_k$ is scalar, therefore its inversion is not a matrix inversion.

Now

$$\mathbf{X}_{N_u}^{-1} = \left[\mathbf{G}^{\mathrm{T}}\Gamma\mathbf{G} + \Lambda\right]^{-1} \quad \text{therefore} \quad \mathbf{u} = \mathbf{X}_{N_u}^{-1}\mathbf{G}^{\mathrm{T}}\Gamma(\mathbf{w} - \mathbf{f}) \qquad (7.39)$$

where **u** is the required vector of control moves corresponding to the control horizon of N_u.

7.6 Control Horizon Interpolation with Recursion

In section 7.4 an MPC formulation for a control horizon interpolation was presented. Section 7.5 provided a method for control horizon recursion. In this section a new formulation for combined interpolation and recursion of control horizon is developed. For this purpose the control law (7.22 will be used, i.e.

$$\mathbf{u} = (\mathbf{G}^{\#\mathrm{T}}\Gamma^{\#}\mathbf{G}^{\#} + \Lambda^{\#})^{-1}\mathbf{G}^{\#}\Gamma^{\#}(\mathbf{w}^{\#} - \mathbf{f}^{\#}) \qquad (7.22)$$

Where $\mathbf{G}^{\#}$, $\Gamma^{\#}$, $\mathbf{w}^{\#}$, $\mathbf{f}^{\#}$ and $\Lambda^{\#}$ are as defined in (7.19) and $\mathbf{G}_m$ and Λ_m are defined in (7.20) and (7.21) respectively. **G** is the usual dynamic matrix corresponding to a control horizon of N_u as used in section 7.2. The control vector is defined as:

$$\mathbf{u}=\begin{bmatrix}\mathbf{u}^{\#}_{\mu}\\ \mathbf{u}^{\#}_{\mu m}\end{bmatrix}\quad \text{where } \mathbf{u}^{\#}_{\mu} \text{ is of size } N_u \text{ and } \mathbf{u}^{\#}_{\mu m} \text{ is of size } m \tag{7.40}$$

The term $\mathbf{G}^{\#^{\mathrm{T}}}\Gamma^{\#}\mathbf{G}^{\#}+\Lambda^{\#}$ on the RHS of (7.22) is:

$$\begin{aligned}\mathbf{G}^{\#^{\mathrm{T}}}\Gamma^{\#}\mathbf{G}^{\#}+\Lambda^{\#} &= \begin{bmatrix}\mathbf{G}^{\mathrm{T}} & \mathbf{G}^{\mathrm{T}}\\ \mathbf{G}^{\mathrm{T}}_{\mathrm{m}} & 0\end{bmatrix}\begin{bmatrix}\mu\Gamma & 0\\ 0 & (1-\mu)\Gamma\end{bmatrix}\begin{bmatrix}\mathbf{G} & \mathbf{G}_{\mathrm{m}}\\ \mathbf{G} & 0\end{bmatrix}+\begin{bmatrix}\Lambda & 0\\ 0 & \Lambda_{\mathrm{m}}\end{bmatrix}\\ &= \begin{bmatrix}\mathbf{G}^{\mathrm{T}}\Gamma\mathbf{G}+\Lambda & \mathbf{G}^{\mathrm{T}}\mu\Gamma\mathbf{G}_{\mathrm{m}}\\ \mathbf{G}^{\mathrm{T}}_{\mathrm{m}}\mu\Gamma\mathbf{G} & \mathbf{G}^{\mathrm{T}}_{\mathrm{m}}\mu\Gamma\mathbf{G}_{\mathrm{m}}+\Lambda_{\mathrm{m}}\end{bmatrix}=\begin{bmatrix}\mathbf{X} & \mathbf{Z}_{\mu}\\ \mathbf{Z}^{\mathrm{T}}_{\mu} & \mathbf{S}_{\mu}\end{bmatrix}\end{aligned} \tag{7.41}$$

where

$$\mathbf{X}=\mathbf{G}^{\mathrm{T}}\Gamma\mathbf{G}+\Lambda,\quad \mathbf{Z}_{\mu}=\mathrm{G}^{\mathrm{T}}\mu\Gamma\mathrm{G}_{\mathrm{m}},\quad \mathbf{Z}^{\mathrm{T}}_{\mu}=\mathbf{G}^{\mathrm{T}}_{\mathrm{m}}\mu\Gamma\mathbf{G},\ \text{and}\ \ \mathbf{S}_{\mu}=\mathbf{G}^{\mathrm{T}}_{\mathrm{m}}\mu\Gamma\mathbf{G}_{\mathrm{m}}+\Lambda_{\mathrm{m}} \tag{7.42}$$

Applying the matrix inversion lemma gives

$$\begin{gathered}\left[\mathbf{G}^{\#^{\mathrm{T}}}\Gamma^{\#}\mathbf{G}^{\#}+\Lambda^{\#}\right]^{-1}=\begin{bmatrix}\mathbf{X} & \mathbf{Z}_{\mu}\\ \mathbf{Z}^{\mathrm{T}}_{\mu} & \mathbf{S}_{\mu}\end{bmatrix}^{-1}=\begin{bmatrix}\mathbf{X}^{-1}-\mathbf{W}_{\mu}\mathbf{Z}^{\mathrm{T}}_{\mu}\mathbf{X}^{-1} & \mathbf{W}_{\mu}\\ \mathbf{W}^{\mathrm{T}}_{\mu} & \mathbf{V}_{\mu}\end{bmatrix}\\ \text{with } \mathbf{W}_{\mu}=-\mathbf{X}^{-1}\mathbf{Z}_{\mu}\mathbf{V}_{\mu}\quad ,\mathbf{W}^{\mathrm{T}}_{\mu}=-\mathbf{V}^{\mathrm{T}}_{\mu}\mathbf{Z}^{\mathrm{T}}_{\mu}\mathbf{X}^{-1}\quad \text{and}\quad \mathbf{V}_{\mu}=(\mathbf{S}_{\mu}-\mathbf{Z}^{\mathrm{T}}_{\mu}\mathbf{X}^{-1}\mathbf{Z}_{\mu})^{-1}\end{gathered} \tag{7.43}$$

The dimensions of the various matrices are as follows:

$$\begin{array}{llllllll}\mathbf{G} & \text{is} & N\times N_u & \mathbf{G}_{\mathrm{m}} & \text{is} & N\times m & \mathbf{G}^{\#} & \text{is}\ \ 2N\times(N_u+m)\\ \mathbf{G}^{\#} & \text{is} & 2N\times 2N & \Lambda^{*} & \text{is} & (N_u+m)\times(N_u+m) & & \\ \Lambda & \text{is} & N_u\times N_u & \Lambda_{m} & \text{is} & m\times m) & \mathbf{X} & \text{is}\ \ N_u\times N_u\\ \mathbf{Z}_{\mu} & \text{is} & N_u\times m & \mathbf{S}_{\mu} & \text{is} & m\times m & \mathbf{V}_{\mu} & \text{is}\ \ m\times m\\ \mathbf{W}_{\mu} & \text{is} & N_u\times m & & & & & \end{array} \tag{7.44}$$

The term $(\mathbf{G}^{\#^{\mathrm{T}}}\Gamma^{\#}\mathbf{G}^{\#}+\Lambda^{\#})^{-1}\mathbf{G}^{\#}\Gamma^{\#}$ of (7.22) is

$$\begin{aligned}(\mathbf{G}^{\#^{\mathrm{T}}}\Gamma^{\#}\mathbf{G}^{\#}+\Lambda^{\#})^{-1}\mathbf{G}^{\#}\Gamma^{\#} &= \begin{bmatrix}\mathbf{X} & \mathbf{Z}_{\mu}\\ \mathbf{Z}^{\mathrm{T}}_{\mu} & \mathbf{S}_{\mu}\end{bmatrix}^{-1}\begin{bmatrix}\mathbf{G}^{\mathrm{T}} & \mathbf{G}^{\mathrm{T}}\\ \mathbf{G}^{\mathrm{T}}_{\mathrm{m}} & 0\end{bmatrix}\begin{bmatrix}\mu\Gamma & 0\\ 0 & (1-\mu)\Gamma\end{bmatrix}\\ &= \begin{bmatrix}\mathbf{X}^{-1}-\mathbf{W}_{\mu}\mathbf{Z}^{\mathrm{T}}_{\mu}\mathbf{X}^{-1} & \mathbf{W}_{\mu}\\ \mathbf{W}^{\mathrm{T}}_{\mu} & \mathbf{V}_{\mu}\end{bmatrix}\begin{bmatrix}\mathbf{G}^{\mathrm{T}} & \mathbf{G}^{\mathrm{T}}\\ \mathbf{G}^{\mathrm{T}}_{\mathrm{m}} & 0\end{bmatrix}\begin{bmatrix}\mu\Gamma & 0\\ 0 & (1-\mu)\Gamma\end{bmatrix}\end{aligned} \tag{7.45}$$

or

$$(\mathbf{G}^{\#\mathrm{T}}\Gamma^{\#}\mathbf{G}^{\#}+\Lambda^{\#})^{-1}\mathbf{G}^{\#}\Gamma^{\#}$$

$$=\begin{bmatrix}(\mathbf{X}^{-1}-\mathbf{W}_{\mu}\mathbf{Z}_{\mu}^{\mathrm{T}}\mathbf{X}^{-1})\mathbf{G}^{\mathrm{T}}+\mathbf{W}_{\mu}\mathbf{G}_{\mathrm{m}}^{\mathrm{T}} & (\mathbf{X}^{-1}-\mathbf{W}_{\mu}\mathbf{Z}_{\mu}^{\mathrm{T}}\mathbf{X}^{-1})\mathbf{G}^{\mathrm{T}}\\ \mathbf{W}_{\mu}^{\mathrm{T}}\mathbf{G}^{\mathrm{T}}+\mathbf{V}_{\mu}\mathbf{G}_{\mathrm{m}}^{\mathrm{T}} & \mathbf{W}_{\mu}^{\mathrm{T}}\mathbf{G}^{\mathrm{T}}\end{bmatrix}\begin{bmatrix}\mu\Gamma & 0\\ 0 & (1-\mu)\Gamma\end{bmatrix}$$

$$=\begin{bmatrix}(\mathbf{X}^{-1}-\mathbf{W}_{\mu}\mathbf{Z}_{\mu}^{\mathrm{T}}\mathbf{X}^{-1})\mathbf{G}^{\mathrm{T}}\mu\Gamma+\mathbf{W}_{\mu}\mathbf{G}_{\mathrm{m}}^{\mathrm{T}}\mu\Gamma & (\mathbf{X}^{-1}-\mathbf{W}_{\mu}\mathbf{Z}_{\mu}^{\mathrm{T}}\mathbf{X}^{-1})\mathbf{G}^{\mathrm{T}}(1-\mu)\Gamma\\ \mathbf{W}_{\mu}^{\mathrm{T}}\mathbf{G}^{\mathrm{T}}\mu\Gamma+\mathbf{V}_{\mu}\mathbf{G}_{\mathrm{m}}^{\mathrm{T}}\mu\Gamma & \mathbf{W}_{\mu}^{\mathrm{T}}\mathbf{G}^{\mathrm{T}}(1-\mu)\Gamma\end{bmatrix}$$

(7.46)

The control law of (7.22) is given as:

$$\begin{bmatrix}\mathbf{u}_{\mu}^{\#}\\ \mathbf{u}_{\mu\mathrm{m}}^{\#}\end{bmatrix}=(\mathbf{G}^{\#\mathrm{T}}\Gamma^{\#}\mathbf{G}^{\#}+\Lambda^{\#})^{-1}\mathbf{G}^{\#}\Gamma^{\#}\begin{bmatrix}\mathbf{w}-\mathbf{f}\\ \mathbf{w}-\mathbf{f}\end{bmatrix}$$

$$=\begin{bmatrix}(\mathbf{X}^{-1}-\mathbf{W}_{\mu}\mathbf{Z}_{\mu}^{\mathrm{T}}\mathbf{X}^{-1})\mathbf{G}^{\mathrm{T}}\mu\Gamma+\mathbf{W}_{\mu}\mathbf{G}_{\mathrm{m}}^{\mathrm{T}}\mu\Gamma & (\mathbf{X}^{-1}-\mathbf{W}_{\mu}\mathbf{Z}_{\mu}^{\mathrm{T}}\mathbf{X}^{-1})\mathbf{G}^{\mathrm{T}}(1-\mu)\Gamma\\ \mathbf{W}_{\mu}^{\mathrm{T}}\mathbf{G}^{\mathrm{T}}\mu\Gamma+\mathbf{V}_{\mu}\mathbf{G}_{\mathrm{m}}^{\mathrm{T}}\mu\Gamma & \mathbf{W}_{\mu}^{\mathrm{T}}\mathbf{G}^{\mathrm{T}}(1-\mu)\Gamma\end{bmatrix}\begin{bmatrix}\mathbf{w}-\mathbf{f}\\ \mathbf{w}-\mathbf{f}\end{bmatrix}$$

(7.47)

or

$$=\begin{bmatrix}(\mathbf{X}^{-1}-\mathbf{W}_{\mu}\mathbf{Z}_{\mu}^{\mathrm{T}}\mathbf{X}^{-1})\mathbf{G}^{\mathrm{T}}\Gamma(\mathbf{w}-\mathbf{f})+\mathbf{W}_{\mu}\mathbf{G}_{\mathrm{m}}^{\mathrm{T}}\mu\Gamma(\mathbf{w}-\mathbf{f})\\ \mathbf{W}_{\mu}^{\mathrm{T}}\mathbf{G}^{\mathrm{T}}\Gamma(\mathbf{w}-\mathbf{f})+\mathbf{V}_{\mu}\mathbf{G}_{\mathrm{m}}^{\mathrm{T}}\mu\Gamma(\mathbf{w}-\mathbf{f})\end{bmatrix}$$

(7.48)

$$=\begin{bmatrix}(\mathbf{I}+\mathbf{X}^{-1}\mathbf{Z}_{\mu}\mathbf{V}_{\mu}\mathbf{Z}_{\mu}^{\mathrm{T}})\mathbf{X}^{-1}\mathbf{G}^{\mathrm{T}}\Gamma(\mathbf{w}-\mathbf{f})-\mathbf{X}^{-1}\mathbf{Z}_{\mu}\mathbf{V}_{\mu}\mathbf{G}_{\mathrm{m}}^{\mathrm{T}}\mu\Gamma(\mathbf{w}-\mathbf{f})\\ -\mathbf{V}_{\mu}^{\mathrm{T}}\mathbf{Z}_{\mu}^{\mathrm{T}}\mathbf{X}^{-1}\mathbf{G}^{\mathrm{T}}\Gamma(\mathbf{w}-\mathbf{f})+\mathbf{V}_{\mu}\mathbf{G}_{\mathrm{m}}^{\mathrm{T}}\mu\Gamma(\mathbf{w}-\mathbf{f})\end{bmatrix}$$

$$\begin{bmatrix}\mathbf{u}_{\mu}^{\#}\\ \mathbf{u}_{\mu\mathrm{m}}^{\#}\end{bmatrix}=\begin{bmatrix}(\mathbf{I}+\mathbf{X}^{-1}\mathbf{Z}_{\mu}\mathbf{V}_{\mu}\mathbf{Z}_{\mu}^{\mathrm{T}})\mathbf{u}-\mathbf{X}^{-1}\mathbf{Z}_{\mu}\mathbf{V}_{\mu}\mathbf{G}_{\mathrm{m}}^{\mathrm{T}}\mu\Gamma(\mathbf{w}-\mathbf{f})\\ -\mathbf{V}_{\mu}^{\mathrm{T}}\mathbf{Z}_{\mu}^{\mathrm{T}}\mathbf{u}+\mathbf{V}_{\mu}\mathbf{G}_{\mathrm{m}}^{\mathrm{T}}\mu\Gamma(\mathbf{w}-\mathbf{f})\end{bmatrix} \tag{7.49}$$

$$\begin{aligned}\mathbf{u}_{\mu}^{\#}&=(\mathbf{I}+\mathbf{X}^{-1}\mathbf{Z}_{\mu}\mathbf{V}_{\mu}\mathbf{Z}_{\mu}^{\mathrm{T}})\mathbf{u}-\mathbf{X}^{-1}\mathbf{Z}_{\mu}\mathbf{V}_{\mu}\mathbf{G}_{\mathrm{m}}^{\mathrm{T}}\mu\Gamma(\mathbf{w}-\mathbf{f})\\ &=\mathbf{u}+\mathbf{X}^{-1}\mathbf{Z}_{\mu}\mathbf{V}_{\mu}\left[\mathbf{Z}_{\mu}^{\mathrm{T}}\mathbf{u}-\mathbf{G}_{\mathrm{m}}^{\mathrm{T}}\mu\Gamma(\mathbf{w}-\mathbf{f})\right]\end{aligned} \tag{7.50}$$

$$\mathbf{u}_{\mu\mathrm{m}}^{\#}=-\mathbf{V}_{\mu}^{\mathrm{T}}\mathbf{Z}_{\mu}^{\mathrm{T}}\mathbf{u}+\mathbf{V}_{\mu}\mathbf{G}_{\mathrm{m}}^{\mathrm{T}}\mu\Gamma(\mathbf{w}-\mathbf{f})=-\mathbf{V}_{\mu}^{\mathrm{T}}\left[\mathbf{Z}_{\mu}^{\mathrm{T}}\mathbf{u}-\mathbf{G}_{\mathrm{m}}^{\mathrm{T}}\mu\Gamma(\mathbf{w}-\mathbf{f})\right] \tag{7.51}$$

For $\mu = 1$ the above equations reduce to the equations for the recursion-only case. Setting $\mu = 0$ yields $\mathbf{u}^{\#}_{\mu} = \mathbf{u}$ and $\mathbf{u}^{\#}_{\mu m} = 0$, i.e. the base case with neither recursion nor interpolation. Implementation follows the procedure outlined in Section 7.5.

7.7 Simulation Examples

The notion of control horizon interpolation, recursion and interpolation with recursion developed in the preceding sections are demonstrated using Matlab based simulations in the following sub-sections. Examples are designed to illustrate the following:

- Interpolation between N_u and N_u+m using μ-weighting Methods I and II.
- Control horizon recursion.
- μ-interpolation with recursion

All of the simulations are based on the following z-domain 3rd order process (McIntosh, 1988):

$$\text{z - domain transfer function} = \frac{.00768z^{-1} + .02123z^{-2} + .00357z^{-3}}{1 - 1.9031z^{-1} + 1.1514z^{-2} - .2158z^{-3}} \tag{7.52}$$

The poles of the z-domain transfer function are 0.8187, 0.7165 and 0.3679, while the zeros are at -2.586 and -0.1798. One of the zeros is outside the unit circle making the discretized process nonminimum phase (NMP). The gain of the process is 1.

All simulations are carried out for a period of 150 time instants with a setpoint change of unity at time = 0 and a sustained step disturbance of magnitude 0.5 starting at time = 50. A DMC noise/disturbance model is employed in all simulations and N_1=1, N_2=20, λ=0 are fixed. A perfect process model (i.e. model with no MPM) is used in all examples.

7.7.1 Example-1: μ-interpolation of control horizon, Method-I

Figures 7.1 and 7.2 demonstrate interpolation of the control horizon using μ as the interpolating parameter for Method-I. In Figure 7.1, the interpolation is between control horizons of 1 and 2. In this case μ = 0 corresponds to N_u = 1 and μ =1 is the same as N_u =2. As μ is increased towards 1, the response shifts towards that corresponding to N_u =2.

However the effect of μ is almost negligible for μ between 0 and 0.5 or so. The most effective range of μ is usually between 0.9 to 1.0. Figure 7.1 shows response for $\mu = 0$ (or $N_u = 1$), $\mu = 0.7$, $\mu = 0.9$ and $\mu = 1$ (or $N_u = 2$). It is clear that the response gets more active for increasing μ values i.e. increasing control horizon N_u. Figure 7.2 illustrates the same role of μ interpolation between $N_u = 1$ and $N_u = 3$ and can be compared directly with Figure 7.1.

7.7.2 Example-2: μ-interpolation of control horizon, Method-2

This example simply demonstrates that the two methods for μ-interpolation, i.e. Method-I and Method-II, are identical. Figures 7.3 uses a μ value of 0.8, $N_u = 1$ and $m = 1$. The figure clearly shows that the two methods yield identical results for a fixed value of μ and other parameters. However, Method II is preferred because of its computational efficiency resulting from the smaller matrices.

7.7.3 Example-3: Recursion of control horizon

Figures 7.4-7.6 show that the recursive (in terms of control horizon) solution of MPC is identical to the original non-recursive algorithm as proven in section 7.5. Figure 7.4 uses $N_u = 1$ and $m = 1$ to give a controller equivalent to $N_u = 2$. Figure 7.5 uses $N_u = 2$ and $m = 1$ to yield a recursive solution for $N_u = 3$, while the same solution is obtained using $N_u = 1$ and $m = 2$ in Figure 7.6. This example simply demonstrates that the two methods for μ-interpolation, i.e. Method-I and Method-II, are identical and can be used with $m \geq 1$.

7.7.4 Example-4: Recursion with interpolation of control horizon

The μ-interpolation with control horizon recursion is exactly equivalent to μ-interpolation with the non-recursive calculation of **u** as shown in section 7.6. This fact is demonstrated in this example. In Figure 7.7 a μ value of 0.8, $N_u = 1$ and $m = 1$ are used for the simulation. It can be seen that the combination of μ interpolation with control horizon recursion (N_u =1, m = 1 and μ = 0.8) yields the same results as when interpolation and recursion are done separately.

7.8 Conclusions

A new concept of control horizon interpolation is presented. The μ interpolation can be interpreted as the use of non-integer control horizons. Two methods for μ interpolation are formulated. Method II is a more efficient implementation of Method I due to the lower dimensions of the matrices involved. The interpolation can be made between any two integer control horizons N_u and N_u+m, i.e. it is not limited to consecutive control horizons. The matrix inversions required for the control calculation are ($N_u \times N_u$) and ($m \times m$).

A control horizon-recursive solution for MPC is developed which, with the implementation of section 7.5.2, eliminates the need of matrix inversion (N_u =1, m = 1). More than one control horizon step recursion may be performed. A method for combined interpolation and recursion of control horizon is developed which allows multi step control horizon recursion with interpolation. All the results are proven mathematically and demonstrated through simulations.

References:

Clarke, D.W., Mohtadi, C. and Tuffs, P.S., " Generalized Predictive Control - Part I the Basic Algorithm ", Automatica, Vol. 23, No. 2, pp.137-148, 1987a.

Clarke, D.W., Mohtadi, C. and Tuffs, P.S., " Generalized Predictive Control - Part II Extensions and Interpretations ", Automatica, Vol. 23, No. 2, pp. 149-160, 1987b.

Clarke, D.W.," Adaptive Generalized Predictive Control ", Proceedings of the Fourth International Conference on Process Control (CPCIV), Editors Y. Arkun and W. H. Ray, Padre Island, TX, 1991.

Cutler, C.R., "Dynamic Matrix Control - A Computer Control Algorithm ", AIChE National Meeting, Houston, TX, 1979.

De Keyser, M.C., "Basic Principles of Model Based Predictive Control", Proceedings of the First European Control Conference, pp. 1753-1758, 1991.

McIntosh, A.R., "Performance and Tuning of Adaptive Generalized Predictive Control ", M.Sc. thesis, University of Alberta, 1988.

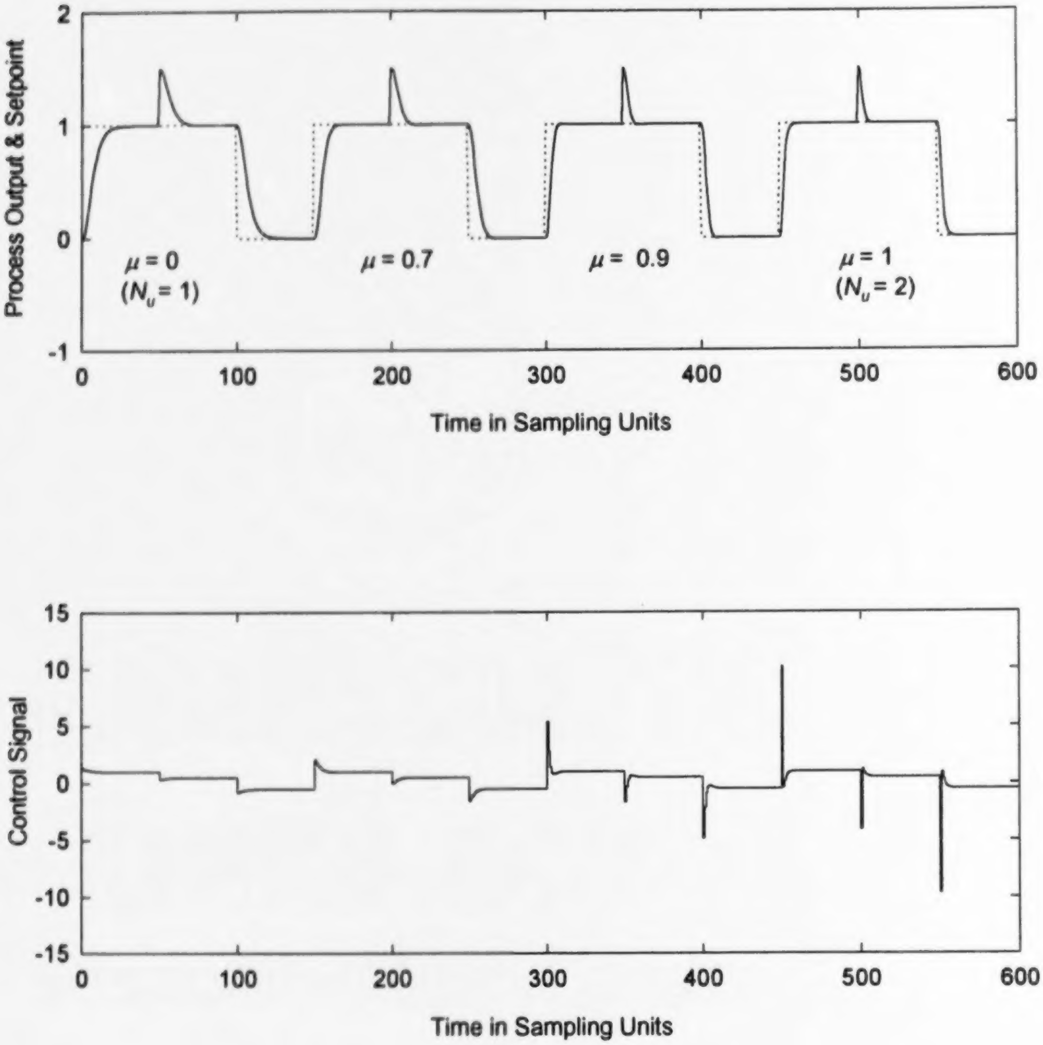


Figure 7.1 μ-weighting interpolation between N_u=1 and N_u=2, Method-I

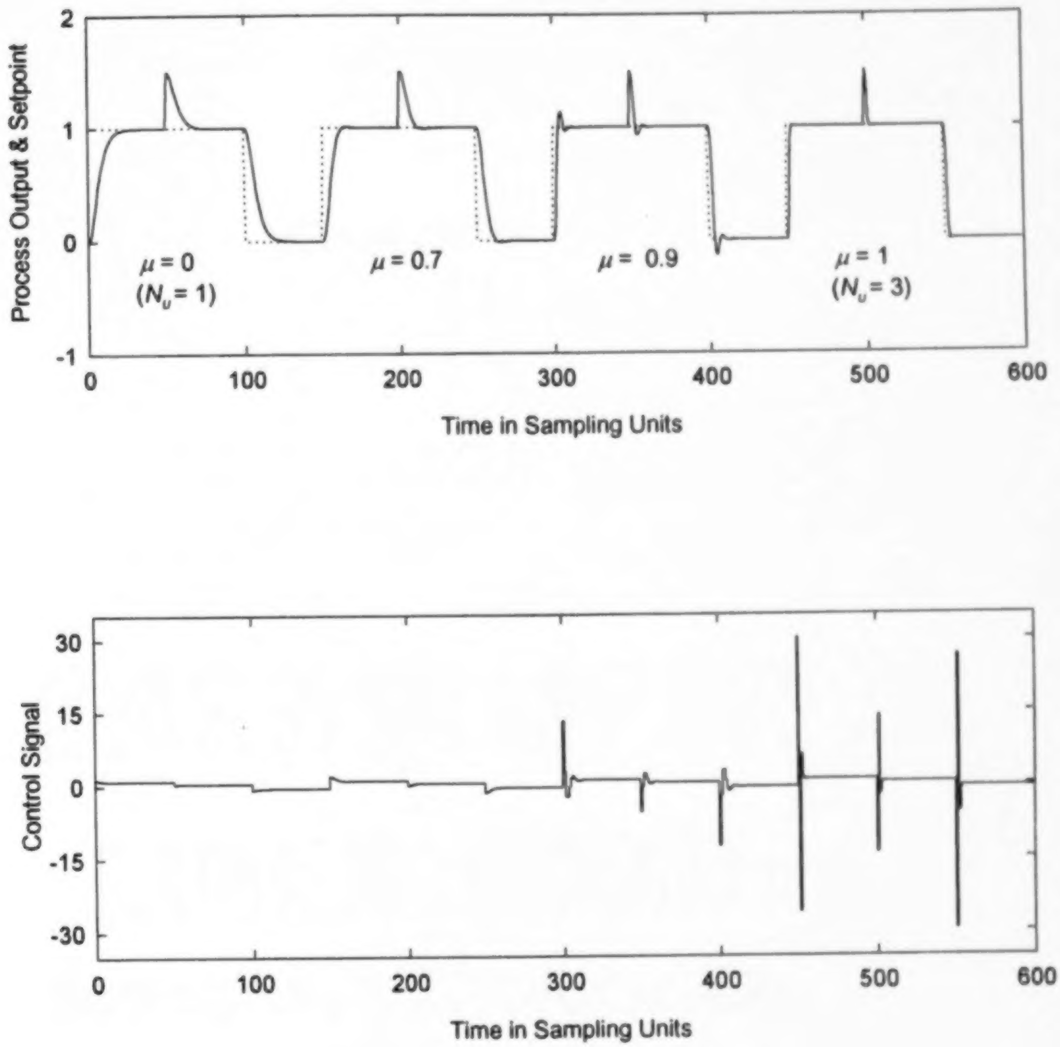


Figure 7.2 μ-weighting interpolation between N_u=1 and N_u=3, Method-I

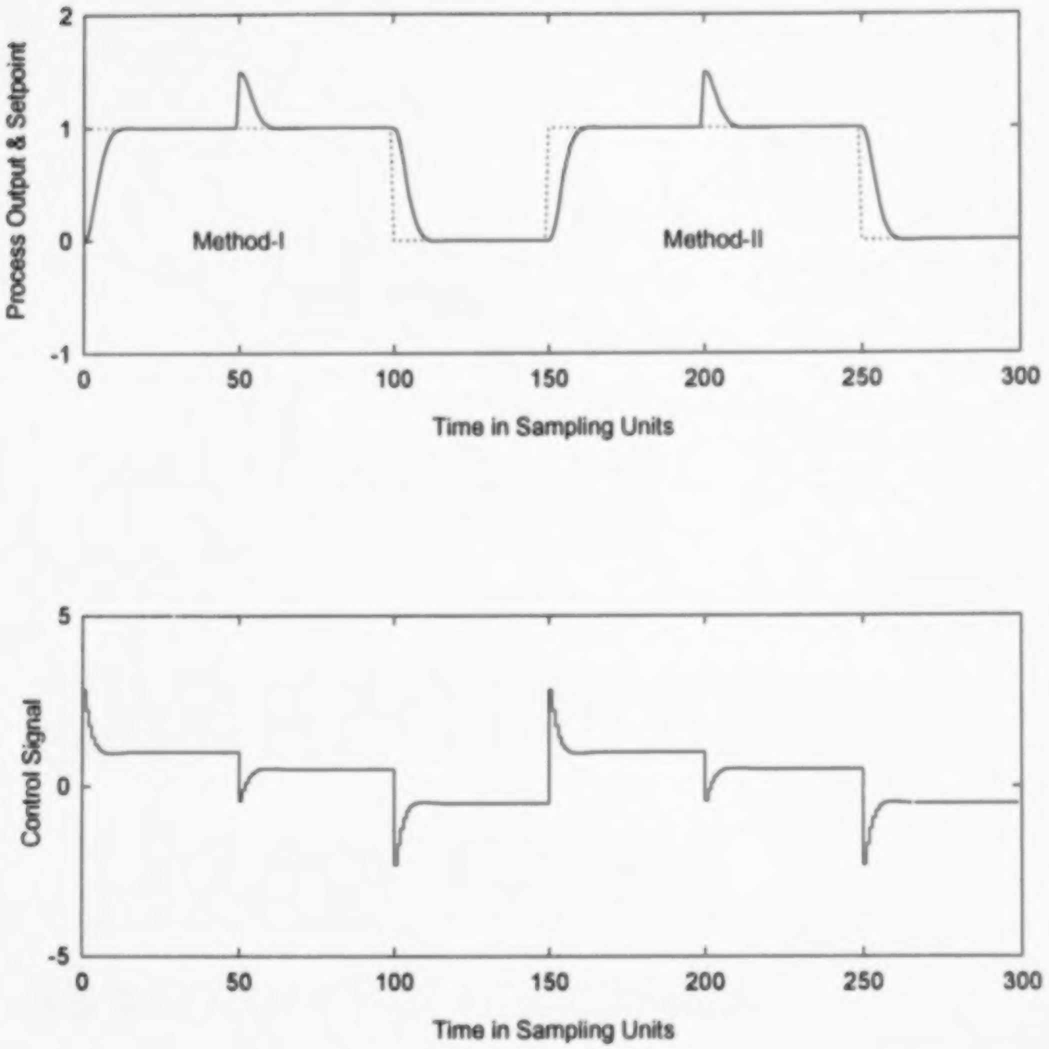


Figure 7.3 μ-weighting Methods I and II, N_u=1, m=1, and μ=0.8

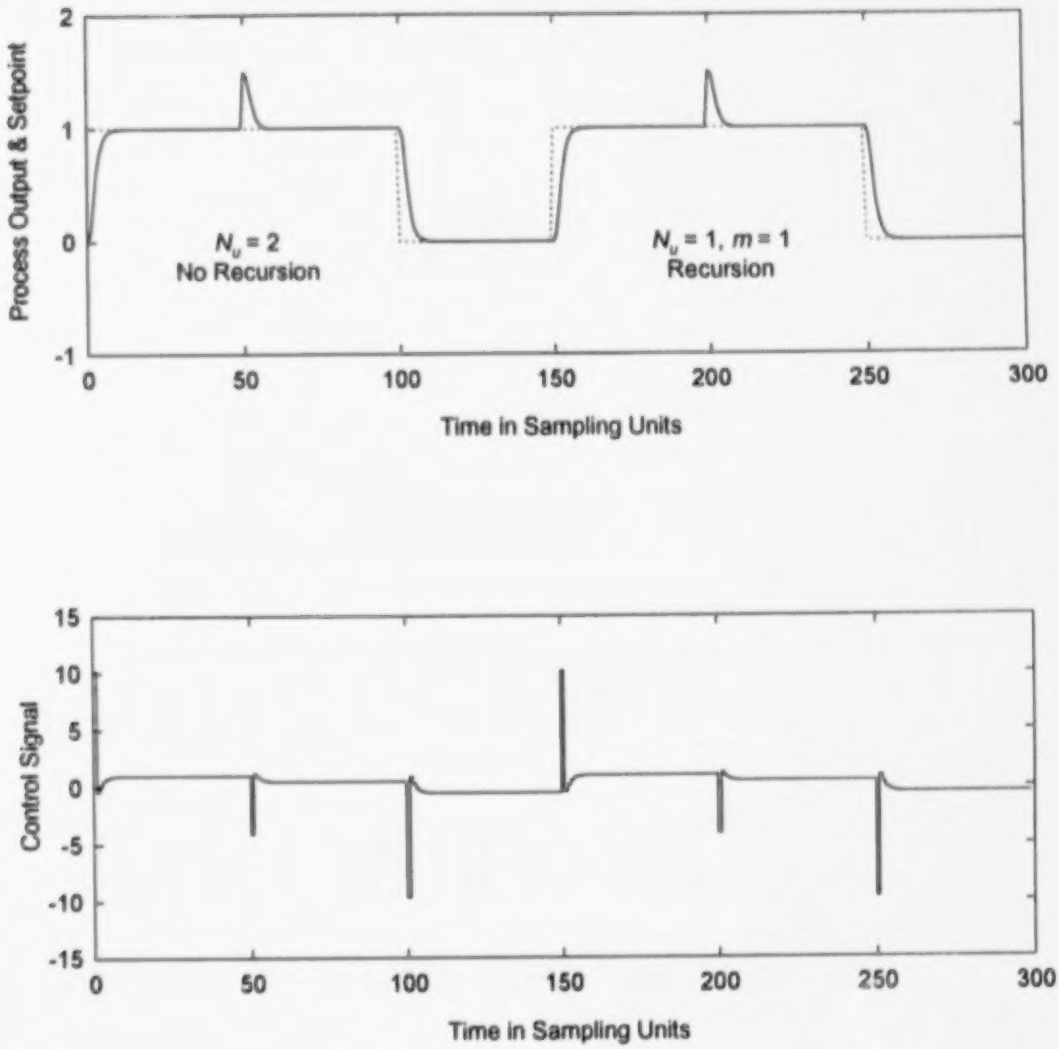


Figure 7.4 Non recursive N_u=2 *vs* recursive N_u=1 with m=1

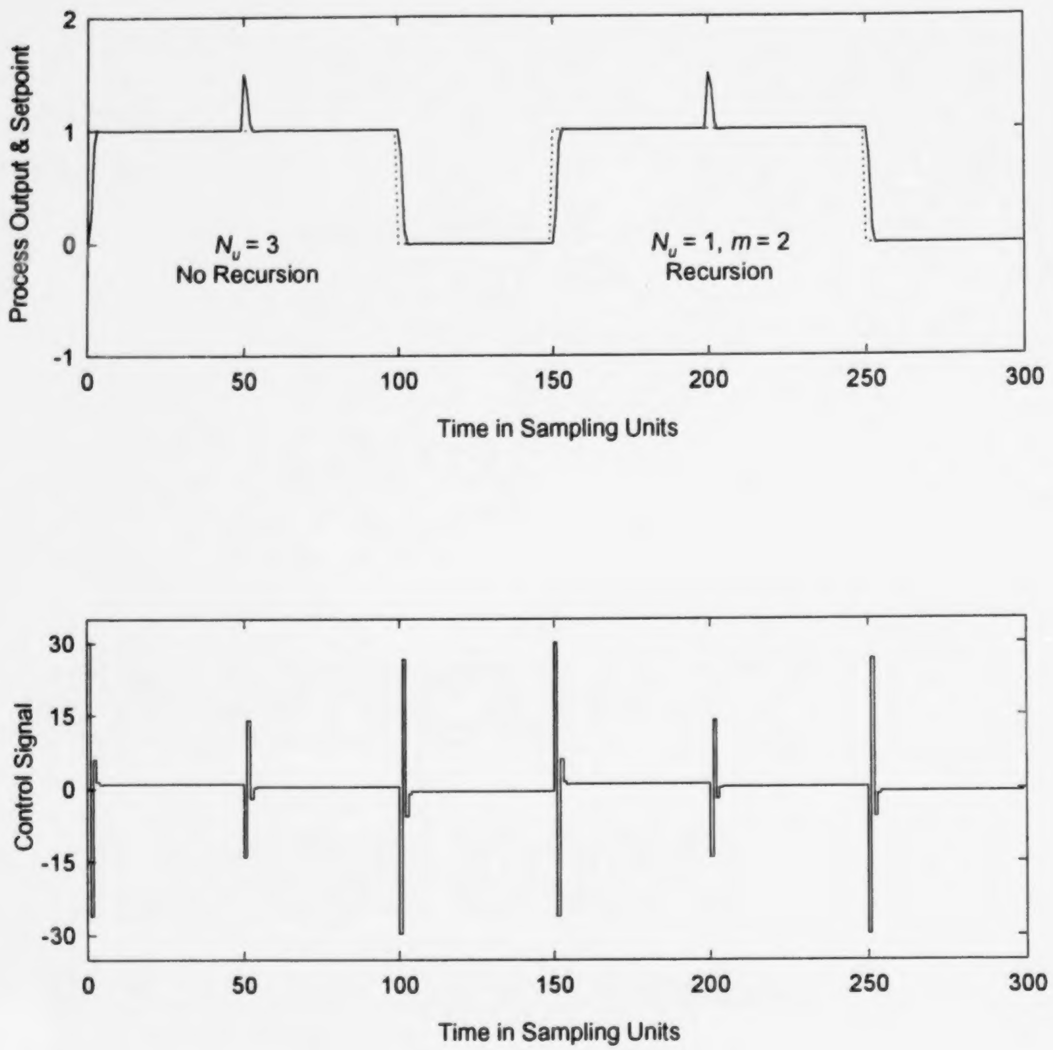


Figure 7.5 Non recursive N_u=3 *vs* recursive N_u=1 with m=2

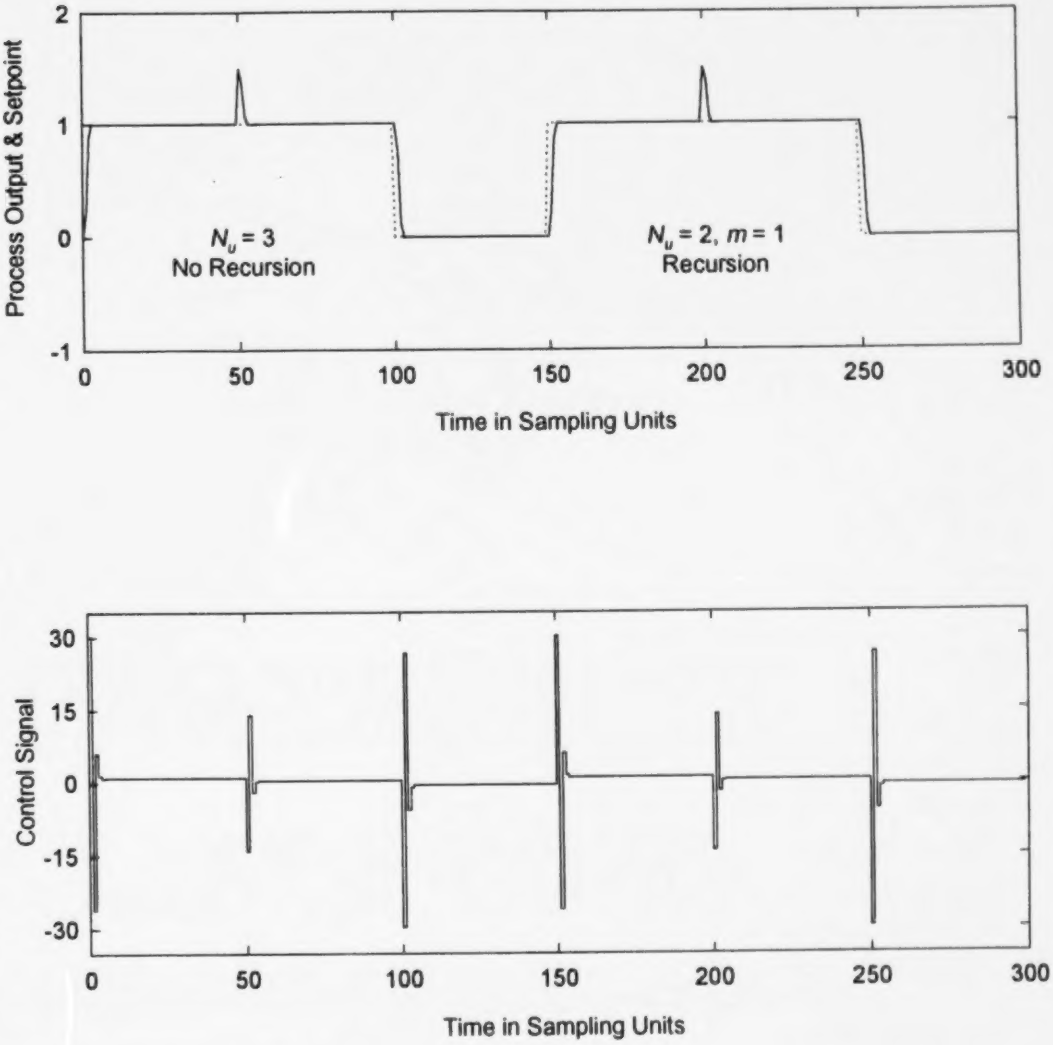


Figure 7.6 Non recursive N_u=3 *vs* recursive N_u=2 with m=1

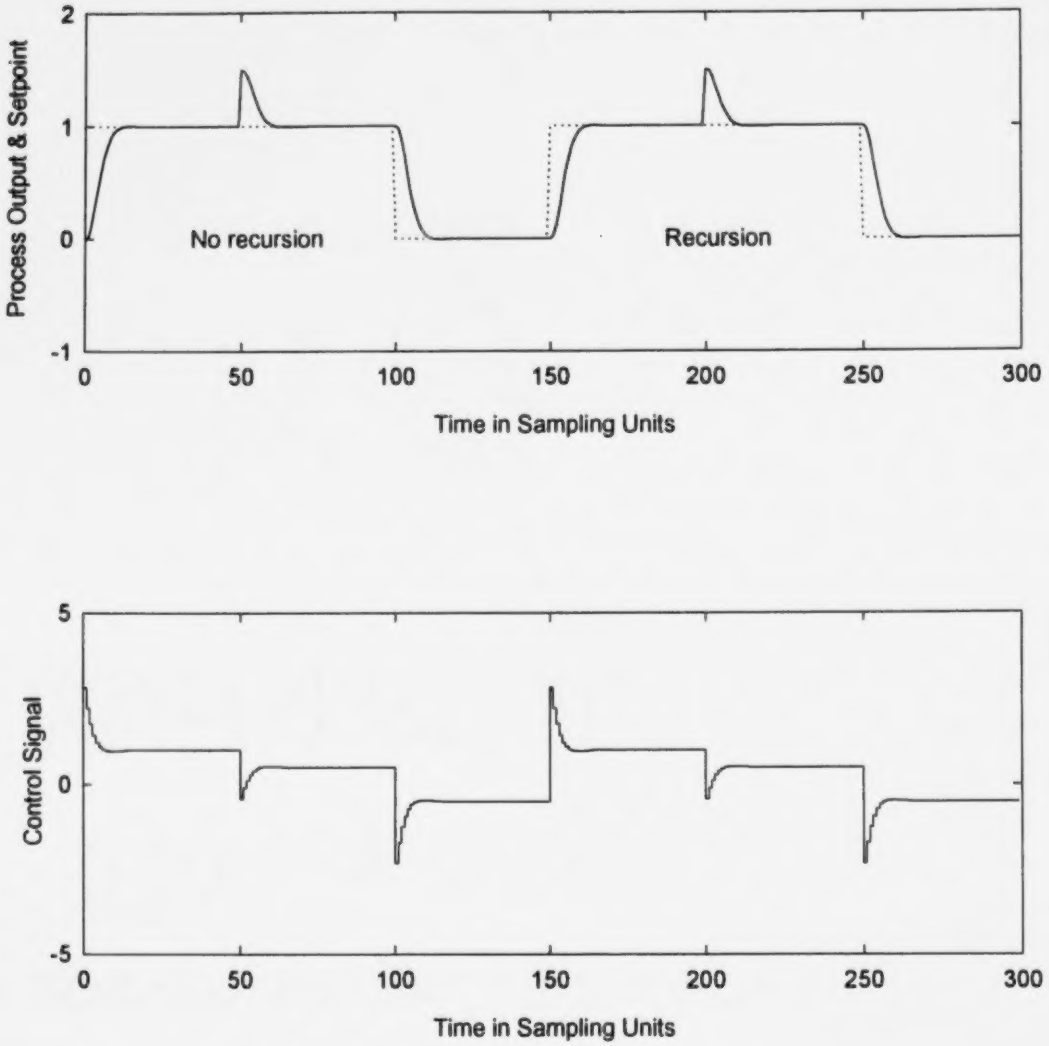


Figure 7.7 μ-interpolation without and with recursion, N_u=1, m=1, and μ=0.8

Chapter 8

Conclusions

8.1 Contributions

A new Unified Model Predictive Controller (UMPC) has been formulated based on the latest methodology available in the literature plus the best combination of the improvements developed in this thesis. The contributions of this research work on MPC are grouped under the following headings and discussed in the following six subsections

1. Generalized noise/disturbance model structure
2. Optimal long-range predictors
3. Use of multiple noise/disturbance models
4. Steady state error weighting
5. Horizon restructuring and interpolation
6. Interpolation and recursion of control horizon

8.1.1 Generalized Noise/Disturbance Model Structure

Most of the formulations in this thesis are based on a generalized Box-Jenkins type model structure which unifies many of the existing MPC algorithms such as GPC and DMC. The generalized model used in UMPC explicitly handles equation error and output error model structures in a single formulation and has provision to incorporate more than one integrator. The role of the noise/disturbance model as a key parameter for disturbance rejection is established and the disturbance rejection properties of GPC and DMC are compared and analyzed in terms of their respective noise/disturbance models.

The classical DMC predictor is derived from the generalized noise/disturbance model and interpreted as a special case of UMPC rather than starting with the classical step response formulation. Based on the generalized noise/disturbance model structure, improved and enhanced versions of DMC (IDMC and EDMC) are formulated. The IDMC noise model includes a first order lag filter in addition to an integrator. This filter gives IDMC an independent parameter to adjust the speed of disturbance rejection. A first order lead-lag filter with an integrator in the noise model of EDMC provides better disturbance rejection and noise attenuation performance than that of classical DMC and can be regarded as an extension of IDMC.

UMPC is formulated using the generalized process and noise models since this gives the user the flexibility to choose the best model for a given application and, in extreme cases, to change the model structure on-line.

8.1.2 Optimal Long-Range Predictors

An optimal long range predictor is a key element of MPC. A considerable portion of the computational effort required for MPC is associated with the long range prediction. Moreover the final mathematical form of MPC is affected by the choice of optimal predictor structure. In the present work a number of different optimal long-range predictors, based on the generalized noise/disturbance model structure, are developed.

A lumped Diophantine predictor or LDP is formulated as a natural extension of the classical optimal predictors used in existing MPC such as GPC (Clarke et. al., 1987a and 1987b). The LDP gives the DMC and GPC predictors as special cases. Overparameterization is used to reduce the computational load and computer storage of long-range predictors. An overparameterized version of LDP termed the lumped Diophantine overparameterized predictor or LDOP is developed to reduce the computational requirements.

The separated Diophantine predictor (SDP) provides the basis for an easier and more straightforward formulation for MPC. The SDP, which maintains separate terms for the process and noise contributions to predictions, offers greater insight in analyzing the role of the noise/disturbance model in long-range predictions. The SDP has been shown to have a Kalman filter (KF) type structure. The separated Diophantine overparameterized predictor (SDOP), an overparameterized version of SDP, offers substantial computational saving. Both predictors, SDP and SDOP, have been proven to be mathematically identical to the original LDP.

The reduced Diophantine predictor (RDP) requires only one set of Diophantine equations in the calculation of the output predictions. The RDOP, an overparameterized form of RDP, yields up to 65% computational saving in adaptive applications but does not separate the process and noise contributions. The reduced Diophantine predictors (RDP and RDOP) have been shown to be mathematically identical to the original LDP.

The Separated Diophantine Overparameterized Predictor (SDOP) was chosen for implementation in UMPC because it is computationally efficient and leads to convenient ways of tuning the disturbance rejection (regulatory control) independent of servo response.

8.1.3 Use of Multiple Noise/Disturbance Models

The noise/disturbance model in an MPC formulation governs the prediction of uncertainty (noise, disturbance and MPM) for future time instants in the output horizon. If a conservative noise model is used the uncertainty contribution to the output prediction does not increase rapidly with time. The DMC noise model, (a random walk structure), is an extreme example of conservative modeling where the uncertainty contribution to the future prediction is set equal to its current value. GPC on the other hand uses an ARIMA noise model in which the increase in the future uncertainty contribution is directly proportional to the "slowness" (or the effective time constant) of the process model.

Aggressive noise models are good for disturbance rejection but are highly non-robust in the presence of unmodeled dynamics.

Conventional MPC employs a single noise/disturbance model for the entire prediction horizon. The present study suggests the use of more than one noise/disturbance model for output predictions. For the case of two noise models the notion of *disturbance horizon* is introduced and defined as the first part of the output horizon which uses the *primary* noise model. The rest of the output horizon employs the *secondary* noise model. Using the disturbance horizon as a tuning parameter, *a flexible controller is developed which can be tuned to give any desired level of disturbance rejection performance between and including DMC and GPC*. A variation of the above approach is to predict the uncertainty contribution using a single noise model over the disturbance horizon and keep it constant afterward rather than use a second noise model. (Note that a disturbance horizon of zero would give DMC).

The disturbance horizon discussed above is a *discrete* (integer) tuning parameter. A non-discrete tuning parameter, σ, is introduced to combine two noise models in any desired proportion. The continuous, normalized tuning parameter, σ, which takes values between 0 and 1, is more convenient for tuning, and its extreme values gives DMC and GPC type performance.

Disturbance horizon and σ-weighting tuning parameters have been incorporated into the MPC algorithm using the separated Diophantine predictors (SDP or SDOP) which allow independent tuning of the uncertainty contribution in the prediction.(The lumped Diophantine predictor (LDP) and the reduced Diophantine predictor (RDP) are not suitable for these extensions.)

Continuous tuning using $0 \leq \sigma \leq 1$ and a SDOP are recommended because of their convenience and computational efficiency. For the ideal case with MPM = 0, σ can be used to *tune the disturbance rejection independent of the servo response*.

8.1.4 Steady State Error Weighting

Many MPC applications use a large prediction horizon in order to provide more robust control. This imposes a higher computational load for adaptive implementations. One solution to this problem is the incorporation of the steady state tracking error in the cost function of MPC. A short output horizon with appropriate steady state weighting can produce the effect of larger output horizons. Steady state weighting in MPC was originally proposed by Kwok and Shah (1992). However, their formulation has several shortcomings as described in Chapter 6. In this thesis steady state weighting (γ_s-weighting) is incorporated into MPC using a completely new formulation. The new formulation is based on the generalized UMPC noise/disturbance model and the separated Diophantine predictor (SDP). The steady state prediction can be calculated using either one of two methods. A single steady state error, corresponding to a control horizon of 1, is used in the new formulation. The use of a single steady state tracking error in the MPC cost function is justified by analysis of long range predictive control involving only the steady state predictions corresponding to control horizons 1 to N_u. The steady state prediction is added as the last element in the vector of predicted outputs. This approach does not involve any extra matrices or vectors and the original form of the MPC control law remains unchanged. The new formulation may be interpreted as an interpolation between the regular MPC and the mean level MPC as the normalized tuning parameter, γ_s, takes the values between 0 and 1. Conservative noise models, e.g. a DMC structure, are recommended for steady state prediction for robustness in the presence of model plant mismatch (MPM).

The γ_s-weighting in UMPC provides a new tuning strategy without any computational penalty. The γ_s-tuning (with appropriate noise model for steady state prediction) is found to be much superior to the λ-weighting and hence is recommended for tuning the combined regulatory/servo response. It is particularly appropriate for systems with unmodelled dynamics if it is combined with σ-weighting approach for disturbance modeling (Section 8.1.3).

8.1.5 Horizon Restructuring and Interpolation

8.1.5.1 Structuring the Output Horizon

The output horizon in classical MPC is a set of contiguous predictions, which starts at the current time t and extends through up to time $t+N_2$. The current study proposes the use of sparse output horizons which contains only arbitrarily selected predictions i.e. a subset from the time interval t to $t+N_2$. A regular spacing of d_1 between the selected predictions for minimization yields results which compare well with the results when there is no spacing, i.e. $d_1 = 0$. The use of $d_1 > 0$ reduces the dimensions of the dynamic matrix and the free response vector and hence cuts down on the computational requirements even for nonadaptive implementations.

The performance deterioration due to large values of d_1, (i.e. for highly sparse output horizons) may be avoided by minimizing the *integral* of the tracking error squares rather than the *summation*, i.e. using a numerical integration formula more sophisticated than Euler's Method.

A sparse output horizon is recommended (along with numerical integration) to save computational resources.

8.1.5.2 Structuring the Control Horizon

The control horizon, i.e. the number of independent control moves permitted in the optimization is inherently an integer tuning parameter. However in many cases especially in the presence of model plant mismatch (MPM), a higher value of control horizon destabilizes the system while a lower value gives sluggish control. An intermediate level of control horizon between two consecutive integer values is often desirable for active and robust control. The concept of independent and dependent control moves in UMPC is extended to produce three different methods of obtaining the effect of non-discrete control horizons.

One useful special case is to introduce spacing in the classical control horizon. Introducing spacing d_2 after the first independent control move, produces results equivalent to interpolation between control horizons 1 and N_u.

Spacing before the last independent control move, d_3, interpolates between control horizons N_u-1 and N_u.

A third alternative for modifying the control horizon is to use a tuning parameter d_4 to generate a weighted combination of two control horizons, N_{u1} and N_{u2} . This provides the effect of an interpolated control horizon $N_{u1} \leq N_u \leq N_{u2}$. The primary control horizon, N_{u1}, is applicable for the interval t to $t+d_4$ of the output horizon while the remainder of the output horizon uses the secondary control horizon, N_{u2}. The d_4 tuning is a control horizon version of the disturbance horizon concept. Setting $d_4 = 0$ is equivalent to using N_{u2} and $d_4 = N_2$ is equivalent to using N_{u1}.

A method for *continuous* interpolation between two levels of control horizon separated by m with $m \geq 1$ is developed using μ-weighting. Two approaches for μ-weighting are formulated. Note that μ-weighting generates an optimal weighted combination of the control actions produced by two different control horizons whereas σ-weighting produces an optimal combination of the control actions generated using two different noise models. They can be used individually or together.

The μ-weighting tuning strategy is recommended because of its continuous range ($0 \leq \mu \leq 1$). It can be used along with σ-weighting to replace the conventional λ-weighting.

8.1.6 Interpolation and Recursion of the Control Horizon

The conventional MPC implementation involves inversion of an N_u -dimensional matrix. For higher values of N_u , matrix inversion poses a substantial computational burden in the case of adaptive implementations. This research work formulates a recursive algorithm for control calculation. The proposed method provides a means to perform *m*-step recursion with $m \geq 1$. For SISO applications with $N_u = 1$ and $m = 1$ no matrix inversion is required. The recursive formulation is combined with the μ-weighting described earlier.

The recursive formulation is recommended as a means of reducing the order of the matrix inversion.

8.2 Tuning Guidelines for UMPC

A systematic way of selecting appropriate tuning options is highly desirable for practical implementations of UMPC. The following objectives should be considered in developing guidelines for the optimal selection of UMPC options.

- Efficient control
- Robust controller design
- Low computational requirements
- Minimum number of tuning parameters
- Simpler Formulation
- Easier implementation

In the following sections separate recommendations are given for open-loop stable and unstable processes.

8.2.1 UMPC for Open-loop Stable Processes

For stable plants the following UMPC options achieve most of the above mentioned performance objectives:

- SDOP for computational savings
- Steady state weighting (γ_s) to detune the servo response without large N_2

- σ-weighting with a performance oriented *primary* noise model (e.g. GPC) and a conservative *secondary* noise model (e.g. DMC) to provide independent tuning of disturbance rejection
- An appropriate number of integrators to give offset-free response
- For poorly damped processes, or for more active control, μ-weighting with $N_u > 1$

Note that σ, γ_s and μ are continuous, normalized tuning parameters.

8.2.2 UMPC for Open-loop Unstable Processes

For unstable plants the following UMPC options are recommended:

- RDOP with an equation error model structure (e.g. ARIMAX)
- C polynomial (T-filter) to improve the robustness of the controller
- An appropriate number of integrators to give offset-free response
- Output horizon spacing to reduce the computational load
- μ-weighting with $N_u > 1$

The minimum prediction horizon, N_1, is always set to be $\geq d + 1$ where d is the process dead time.

Note that the well known λ-weighting is replaced by the μ-weighting.

8.3 Future Work

The work presented in this thesis covers a wide range of topics in MPC. Most of the results in this work were developed based on sound theory and, in most cases, have been related to the existing established formulations. Nevertheless, a number of issues, particularly related to implementation, remain unaddressed and are potential areas for future work. Some of the areas of interest are described below:

- In the present work the generalized noise/disturbance model has been used exclusively as a design element for *tuning* the disturbance rejection performance. On-line estimation of the noise model would provide a more up to date description of the system. Identification algorithms for Box-Jenkins type models are already available in the literature (e.g. the prediction error method or PEM, Ljung, 1987, Sodestrom, 1989). However the issues of convergence and adaptation in real time implementations need further consideration.

- The separated Diophantine predictor (SDP) was shown to have a Kalman filter type structure. Further study to establish a more precise equivalence of the SDP to KF is justified. The key aspects to proceed in this direction are a state space equivalent of the input/output transfer function and the matrix solution of the Diophantine equations.

- More guidelines need to be developed for using the noise/disturbance model as a design parameter in practical applications.

- It should be possible to develop a steady state weighting equivalent formulation for unstable process models, e.g. integrating processes.

- The incorporation of input, output and auxiliary constraints into the UMPC formulations developed in this thesis is highly desirable.

- All of the work in this thesis is based on SISO (single input single output) systems. The results may be extended to MIMO (multi input multi output) systems. However considerations like time delays and interactions have to be analyzed in greater detail for MIMO case.

References:

Clarke, D.W., Mohtadi, C. and Tuffs, P.S., " Generalized Predictive Control - Part I the Basic Algorithm ", Automatica, Vol. 23, No. 2, pp.137-148, 1987a.

Clarke, D.W., Mohtadi, C. and Tuffs, P.S., " Generalized Predictive Control - Part II Extensions and Interpretations ", Automatica, Vol. 23, No. 2, pp. 149-160, 1987b.

Clarke, D.W.," Adaptive Generalized Predictive Control ", Proceedings of the Fourth International Conference on Process Control (CPCIV), Editors Y. Arkun and W. H. Ray, Padre Island, TX, 1991.

Kwok, K., and Shah, S. L., "Long Range Predictive Control with a Terminal Matching Condition", Chemical Engineering Science, Vol. 49, No. 9, pp. 1287-1300, 1994.

Ljung, L., " System Identification: Theory for the User ", Prentice Hall, Englewood Cliffs, New Jersey, 1987.

Soderstrom, T. and Stoica, P., " System Identification", Prentice Hall International Series in Systems and Control Engineering, New York, 1989.